Universidad
Carlos III de Madrid

DOCTORAL THESIS

Gaussian Processes Methods for Nonstationary Regression

Author:

Luis Muñoz González

Supervisors:

Dr. Aníbal R. Figueiras Vidal

Dr. Miguel Lázaro Gredilla

Dpto. de Teoría de la Señal y Comunicaciones

Leganés, Septiembre 2014

TESIS DOCTORAL

GAUSSIAN PROCESSES METHODS FOR NONSTATIONARY REGRESSION

Autor:
LUIS MUÑOZ GONZÁLEZ

Directores:
Dr. ANÍBAL R. FIGUEIRAS VIDAL
Dr. MIGUEL LÁZARO GREDILLA

Firma del Tribunal Calificador:

Firma

Presidente:

Vocal:

Secretario:

Calificación:

Leganés, de de

RESUMEN

Los Procesos Gaussianos (Gaussian Processes, GPs) son una herramienta Bayesiana no paramétrica potente para acometer problemas de regresión no lineal. Como es común para la mayoría de métodos de regresión, los GPs modelan las observaciones como la suma de una función (latente) desconocida y un ruido gaussiano. A diferencia de otras técnicas de regresión, los GPs proceden desde un punto de vista puramente Bayesiano, infiriendo la probabilidad a posteriori de la función desconocida a través de la verosimilitud y la distribución gaussiana que a priori se establece sobre dicha función. Una de las ventajas de los GPs es que proporcionan predicciones probabilísticas – es decir, valores promedio y de dispersión – de forma natural. Por otro lado, normalmente utilizan un número de hiperparámetros reducido, lo que los hace resistentes a problemas de sobreajuste, permitiendo a su vez seleccionar dichos hiperparámetros mediante una sencilla optimización continua de la evidencia. Desafortunadamente, los GPs no pueden ser utilizados para grandes conjuntos de datos, ya que escalan en el tiempo en la forma $\mathcal{O}(N^3)$, limitando su ámbito de aplicación a conjuntos de datos con unos pocos miles de muestras de entrenamiento (utilizando ordenadores de sobremesa actuales), aunque existen aproximaciones dispersas que permiten utilizar los GPs en conjuntos de datos más grandes.

El GP estándar para regresión se formula bajo hipótesis estacionarias: se considera que la potencia de ruido es constante e independiente de los datos de entrada y la función de covarianza de la distribución a priori típicamente depende de la diferencia entre las muestras de entrada. Estas suposiciones puede ser demasiado restrictivas y poco realistas para muchas aplicaciones reales.

Buscando un modelado no estacionario, en esta Tesis proponemos un modelo GP divisivo (Divisive GP, DGP) en el que se combinan dos GPs para lograr no estacionariedad en amplitud y modelado de ruido heterocedástico. La probabilidad a posteriori del modelo divisivo no es tratable de forma analítica, lo que hace necesario proponer algoritmos de inferencia aproximada o métodos tipo Markov Chain Monte Carlo (MCMC) para poder hacer inferencia en dicho modelo. Una de las ventajas

del modelo divisivo es que la verosimilitud es log-cóncava, lo que hace que la distribución a posteriori sea unimodal cuando se combina con una distribución a priori gaussiana. Esto favorece la convergencia de algoritmos de inferencia aproximada como Expectation Propagation (EP) o el método de Laplace.

En primer lugar proponemos EP-DGP, que utiliza una aproximación EP a la distribución a posteriori en el modelo DGP. Los resultados experimentales muestran la buena calidad de dichas aproximaciones comparadas con una implementación MCMC aplicando el algoritmo Elliptical Slice Sampling (ESS) para el mismo modelo, aunque el coste computacional de la aproximación EP es considerablemente menor. Los resultados experimentales en distintas bases de datos – homocedásticas y heterocedásticas – muestran las mejoras del método propuesto con respecto a los métodos del estado del arte en regresión heterocedástica con GPs, así como al propio GP estándar para regresión.

Sin embargo, la carga computacional del EP-DGP es alta comparada con el GP estándar y otras aproximaciones variacionales similares para regresión heterocedástica con GPs. Por este motivo, proponemos también utilizar el método de Laplace para hacer inferencia en el modelo DGP. Las características de la verosimilitud hacen que la distribución a posteriori tenga una forma bastante gaussiana, lo que permite que el método de Laplace proporcione unas aproximaciones a la distribución a posteriori tan precisas como las del EP-DGP, pero a un coste reducido.

Finalmente, también hemos utilizado la aproximación de Laplace para hacer inferencia en un modelo GP para predicción de volatilidad en series temporales financieras, que resulta de aplicación directa para los métodos de regresión heterocedástica. La función de covarianza de Ornstein-Uhlenbeck modela de forma adecuada el comportamiento de este tipo de series financieras y permite que la implementación del método de Laplace escale linealmente con el número de muestras de entrenamiento. Como en el caso anterior, las características de la verosimilitud hacen que las aproximaciones del método de Laplace sean también precisas, en este caso, comparadas con las que proporcionan métodos MCMC aplicados so-

bre el mismo modelo, pero a un coste computacional menor. Los resultados experimentales corroboran las buenas prestaciones del método de Laplace propuesto comparado con otros algoritmos GP similares, reduciendo la carga computacional y proporcionando una capacidad de predicción superior a la de los modelos Generales AutoRegresivos Heterocedásticos (Generalized AutoRegressive Heteroscedastic models, GARCH) comúmente utilizados en tareas de predicción de volatilidad en el campo de la Econometría.

ABSTRACT

Gaussian Processes (GPs) are a powerful nonparametric Bayesian tool for nonlinear regression. As it is common in most regression approaches, GPs models observations as the sum of some unknown (latent) function plus Gaussian noise. Unlike other regression methods, GPs proceed in a purely Bayesian fashion to infer the posterior distribution of the unknown function through the likelihood and a Gaussian prior distribution placed over this unknown function. One of the strengths of GPs is that they produce probabilistic predictions, i.e., average and dispersion values, in a natural way. On the other hand, they usually employ a reduced number of hyperparameters, that can be tuned with a simple continuous optimization of the evidence: This makes them resilient to overfitting. Unfortunately, GPs cannot be applied to large-scale data sets due to their $\mathcal{O}(N^3)$ time scalability, limiting the scope of application to data sets with a few thousands samples (using present desktop computers), although sparse approximations allow to use GPs in bigger data sets.

The standard GP regression is formulated under stationarity hypotheses: The noise power is assumed constant throughout the input space and the covariance of the prior distribution is typically modeled as dependent only on the difference between input samples. This stationary assumption can be too restrictive and unrealistic for many real-world applications.

Pursuing nonstationarity, in this Thesis we propose a Divisive GP (DGP) model, where two GPs are combined to achieve amplitude nonstationarity and heteroscedastic regression. The posterior of the DGP model is analytically intractable, so that approximate inference techniques or Markov Chain Monte Carlo (MCMC) methods are needed to make inference on the model. One of the advantages of the DGP model is that the likelihood is log-concave, which leads to a unimodal posterior when combined with a GP prior. This favors the convergence of approximate inference algorithms as Expectation Propagation (EP) or the Laplace method.

We first propose EP-DGP, an EP posterior approximation to make inference on the DGP model. The experimental results show the high quality of the EP posterior

approximation compared to an MCMC implementation using Elliptical Slice Sampling (ESS) for the same model, but at a reduced cost. The experimental results on different (homoscedastic and heteroscedastic) data sets show the improvements of the proposed method compared with the state-of-the-art methods in heteroscedastic GP regression and the standard GP.

However, the computational burden of EP-DGP is high compared to the standard GP or other similar variational approximations for heteroscedastic regression. We also propose to use the Laplace approximation for the DGP model. The characteristics of the likelihood make the posterior have quite a Gaussian shape, which allows that the Laplace approximation (L-DGP) provides accurate posterior approximations, as in the case of EP-DGP, but at a reduced cost.

Finally, we have also applied the Laplace approximation to make inference on a GP model for volatility forecasting in financial time series, which is a direct application of heteroscedastic regression methods. The use of the Ornstein-Uhlenbeck covariance function, suitable to model the behavior of this kind of time series, allows the Laplace implementation to scale linearly with the number of samples. As in the case of L-DGP, the characteristics of the likelihood make the Laplace approximation an accurate inference procedure, but at a reduced computational load, compared to the MCMC method applied to the same volatility model. The experimental results corroborate the good performance of the Laplace method compared to other similar GP algorithms, reducing the computational burden, and showing better prediction capabilities than the commonly used Generalized AutoRegressive Conditional Heteroscedastic (GARCH) models in volatility forecasting.

“ Nada es tan difícil
que no pueda conseguir la fortaleza”

Julio César

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter we briefly review the basic concepts of Machine Learning and the fundamental types of Learning Machines (LMs) and machine ensembles. This review pretends to introduce the general framework where this Thesis is contextualized. Then, we discuss the motivation and the objectives of the Thesis, which focuses on the development of nonstationary and heteroscedastic regression algorithms using Gaussian Processes (GPs), including an application to volatility forecasting in financial time series. Finally, we shortly describe the contents of the rest of the chapters.

1.1 Learning machines

Fundamentally, machine learning algorithms are oriented to solve two problem families:

- Decision problems, in which, given a set of observations (variables, features or characteristics), we have to decide among a finite number of alternatives. In this family we include the hypotheses test, detection, and classification problems.

- Estimation problems, where a value for a continuous magnitude that cannot be observed, is proposed given a set of features. Here, we include prediction, interpolation, and extrapolation tasks.

Solving these problems by humans goes back to the origin of our species and, many of them, are related to survival, so that all human beings face them. Although the human Evolution has provided capabilities to solve many of them, the quality of our solutions is not satisfactory in some cases, as discussed in [Myers, 2004, Kahneman, 2011]. For example, under hostile or rapidly changing environments, we are not able to deal with these problems properly. Then, it is useful to design and use algorithms that propose solutions to solve or help humans solving decision or estimation tasks, justifying the usefulness and benefit of LMs.

An LM aims to determine a target t, continuous for estimation problems or discrete for decision tasks, observing a set of features $\mathbf{x}$ by means of a function $f_{\mathbf{w}}(\mathbf{x})$, where $\mathbf{w}$ is a set of parameters to be determined. These parameters are tuned in order to statistically minimize a cost function $C(t, f_{\mathbf{w}})$, averaged on a set of training observations. As pointed out in [Figueiras Vidal, 2013], this process is similar to human learning, where we can learn the function f from our perceptions $\mathbf{x}$ and our memory $\mathbf{w}$. In this case the cost C is related to personal assessments (including principles, emotions, and feelings) of the errors. For the case of LMs, we can provide the machine a set of labeled examples (with the correct answers) $\{\mathbf{x}_n, t_n\}$, $n = 1, ..., N$, to evaluate the cost function.

1.1.1 Neural networks

One of the main challenges in machine learning consists on selecting a proper set of functions $f_{\mathbf{w}}$. The first approach was linear regression, addressed at the begining of the XIX Century [Gauss, 1821], where the targets are predicted as linear combina-

tions of the inputs, i.e., $\mathbf{w}^T\mathbf{x} + w_0$, selecting the weights $\mathbf{w}$ and w_0 that minimize the mean squared error given a set of observations. However, this linear model is limited and it is not suitable for most regression problems, where the reality is more complex.

In the case of linear decision algorithms the solution is more demanding. For example, for a binary decision problem, i.e., $t \in \{-1, 1\}$, a linear classifier has to calculate $o(\mathbf{x}) = \text{sgn}(\mathbf{w}^T\mathbf{x} + w_0)$ to determine t, and there is no analytical method to solve the resulting equations. The first formal approach to face this difficulty was proposed by Frank Rosenblatt in [Rosenblatt, 1958], inspired on the ideas of Alan Turing [Turing, 1992], Warren McCulloch and Walter Pitts [McCulloch and Pitts, 1943]. Rosenblatt proposed a linear decision method known as the Perceptron Rule based on the concept of Hebbian learning [Hebb, 1949]. These ideas were known later under the generic concept of reinforcement learning [Sutton and Barto, 1998]. However, the limitations of the model, the algorithmic difficulties and the (mistaken) arguments of Marvin Minsky and Seymour Papert [Minsky and Papert, 1969] virtually close the research in this field for about 30 years.

Meanwhile, the logistic regression was developed in the field of Statistics, where the estimated posterior probabilities of the targets (in the binary classification problem) are given by

$$o(\mathbf{x}) = \text{sgm}(\mathbf{w}^T\mathbf{x} + w_0) = \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{x} - w_0)} \tag{1.1}$$

for targets of the form $t \in \{0, 1\}$ (whereas for ± 1 targets the hyperbolic tangent function is used, instead the sigmoid). The advantage of this formulation is that conventional minimization algorithms as gradient descent can be directly applied to find the weights $\mathbf{w}$ and w_0.

This formulation allows to stack layers of elementary single layer perceptrons, which is the basis of Multi-Layer-Perceptrons (MLPs). This architecture is capable to solve nonlinear (decision and regression) problems and it can be trained

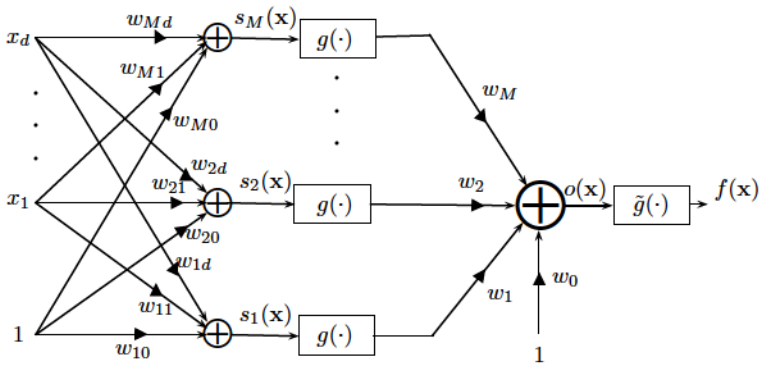


Figure 1.1: Architecture of an MLP with a single hidden layer and one target output.

using the Back-Propagation (BP) algorithm, a form of the chain rule for the derivatives of the training errors. BP was proposed repeteadly [Werbos, 1974, Parker, 1982, LeCun, 1985]. However, the algorithm was not popularized until 1986 thanks to David Rumelhart and his colleagues at MIT [Rumelhart et al., 1986a, Rumelhart et al., 1986b]. This kind of architectures is known as artificial Neural Networks (NNs) given its similarities with the biological neural networks.

In Figure 1.1 we present the architecture of a MLP with a single layer and a scalar output, where the functions $g(\cdot)$ and $\tilde{g}(\cdot)$ are usually sigmoid or hyperbolic tangent functions, although $\tilde{g}(\cdot)$ can also be linear for the case of regression problems. As it can be appreciated in Figure 1.1, the MLP can be considered as a semilinear system, i.e., inputs are transformed nonlinearly in the first layer, whereas the final step is linear.

Following a similar approach, Radial Basis Function NNs (RBFNNs), proposed in [Powell, 1987] to perform function interpolation, apply local transformations in the input layer, for example according to the Gaussian form $o_i(\mathbf{x}) = \exp(-\gamma_i \|\mathbf{x} - \mathbf{c}_i\|_2^2)$, where $\{\mathbf{c}_i\}$ are known as centroids and $\{\gamma_i\}$ are the precisions (inverse of the variances) of the corresponding Gaussians. After this local transformation, as in the case of the MLPs, the output layer is linear.

RBFNNs and one hidden layer MLPs allow to design universal approximators [Cybenko, 1989, Hornik et al., 1989]. However, the corresponding proofs are not constructive and the number of units (or neurons) in the hidden unit is not determined. Then, the number of training samples and the number of features impose a limit in the number of parameters (weights) of the NNs. So that, other possible architectures with more hidden layers cannot be discarded.

The design of Deep NNs (DNN) has gained importance and popularity during the past years. Research on this field has opened different approaches:

- Convolutional NNs (CNNs) [Fukushima, 1979, LeCun et al., 1989] propose to share weights for equivalent regions in the input space, which is suitable for speech or image processing applications, for example.

- Deep Auto-Enconders (DAEs) propose to construct the hidden layers of the deep network step by step aiming to reconstruct the training inputs at each output [Bengio et al., 2007, Vincent et al., 2008].

- Deep Belief Networks (DBNs) [Hinton and Salakhutdinov, 2006], [Hinton et al., 2006], are stacked Restricted Boltzmann Machines (RBMs), which are simplified versions of the Boltzmann Machines (BMs) introduced in [Hinton and Sejnowski, 1986]. BMs are constituted by interconnected input, hidden, and output units where the probabilities of the states of the network are determined by an energy function which depends on the weights of the network.

A more complete description on NNs can be found in [Haykin, 2007]. For complete survey on DNNs, see [Schmidhuber, 2014].

1.1.2 Kernel methods

Kernel methods propose alternative transformations to the inputs using the theorem of representation or kernel trick [Aizerman et al., 1964, Kimeldorf and Wahba, 1971]. The objective of kernel methods is to assume a set of transformations $\phi(\mathbf{x})$ to allow a linear formulation on the transformed space, where the inner products $\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$ are substituted by Mercer kernels, $k(\mathbf{x}_i, \mathbf{x}_j)$, [Schölkopf and Smola, 2001].

This category includes GPs [Williams and Rasmussen, 1996], where a prior distribution over functions is imposed defining a mean function (usually assumed zero) and a covariance function, which needs to be a valid Mercer kernel. GPs proceed in a Bayesian framework inferring the posterior probability over the unknown latent function given the prior and the likelihood of the model (which assumes Gaussian noise in the standard case). In Chapter 2 we provide a more detailed description of GPs.

On the other hand, the Support Vector Machines (SVMs) [Boser et al., 1992] emerged from the work of Vladimir Vapnik in classification problems [Vapnik, 1982], proposing a maximum margin formulation, where the margin is defined as the product of the target and the output provided by the machine, i.e., $t\ o(\mathbf{x})$. Then, SVMs propose kernel classifiers designed to provide the optimal hyperplane (in a classification task) in a high dimensional space $\mathcal{H}$ (also known as feature space), generated from a function transformation $\phi(\cdot) : \mathcal{R}^d \rightarrow \mathcal{H}$ [Vapnik, 1999]. In general, the solution of the SVM is expressed using the discriminant function:

$$f(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b \tag{1.2}$$

where $\mathbf{w}$ and b are the weights that determine the classification hyperplane in the feature space. In order to obtain a maximum margin solution, these weights can be

calculated solving the following optimization problem

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{l=1}^{L}\xi^{(l)} \\ \text{s.t.} \quad & t^{(l)}\left[\mathbf{w}^T\phi(\mathbf{x}^{(l)}) + b\right] \geq 1 - \xi^{(l)}; \qquad l = 1,\ldots,L \\ & \xi^{(l)} \geq 0; \qquad\qquad\qquad\qquad\quad l = 1,\ldots,L \end{aligned} \tag{1.3}$$

where C is a positive parameter that controls the tradeoff between the simplicity of the model and the classification error, and $\{\xi^{(l)}\}_{l=1}^{L}$ is the set of slack variables introduced to allow some samples to be misclassified. Since the functional of the optimization problem in (1.3) is convex, the solution is unique.

To solve (1.3), we need to use Lagrange multipliers, $\{a^{(l)}\}_{l=1}^{L}$, and apply Karush-Kuhn-Tucker conditions [Karush, 1939, Kuhn and Tucker, 1951]. The formulation leads to the following Quadratic Programming (QP) problem:

$$\begin{aligned} \max_{\mathbf{a}} \quad & -\frac{1}{2}\sum_{l=1}^{L}\sum_{l'=1}^{L} a^{(l)}a^{(l')}t^{(l)}t^{(l')}K(\mathbf{x}^{(l)},\mathbf{x}^{(l')}) + \sum_{l=1}^{L} a^{(l)} \\ \text{s.t.} \quad & \sum_{l=1}^{L} a^{(l)}t^{(l)} = 0 \\ & C \geq a^{(l)} \geq 0; \qquad l = 1,\ldots,L \end{aligned} \tag{1.4}$$

where K is the kernel associated to $\phi(\cdot)$. Then, the discriminant function can be written as

$$f(\mathbf{x}) = \sum_{l=1}^{L} a^{(l)}t^{(l)}K(\mathbf{x}^{(l)},\mathbf{x}) + b \tag{1.5}$$

where $\{a^{(l)}\}_{l=1}^{L}$ are positive real values which corresponds to the QP problem solution. The points for which $a^{(l)} > 0$ are known as Support Vectors (SVs) and refer to samples that are inside the margin or misclassified.

Finally, to determine the parameter b, we only consider the set of SVs that satisfies the following condition:

$$t^{(l)}\left(\sum_{l'\in S} a^{(l')}t^{(l')}K(\mathbf{x}^{(l)},\mathbf{x}^{(l')}) + b\right) = 1 \tag{1.6}$$

where S denotes the set of indexes of the SVs. Then, b is calculated as

$$b = \frac{1}{N_{S'}} \sum_{l \in S'} \left(y^{(l)} - \sum_{l' \in S} a^{(l')} t^{(l')} K(\mathbf{x}^{(l)}, \mathbf{x}^{(l')}) \right) \tag{1.7}$$

where S' is the subset of samples for which $0 < a^{(l)} < C$.

For regression tasks, Support Vector Regression (SVR) was proposed in [Drucker et al., 1997], where, similarly to SVM, the model produced by SVR depends only on a subset of the training samples, ignoring the training data that are close to the model prediction (within a given threshold ϵ).

1.1.3 Machine ensembles

LM ensembles have been widely applied to improve the performance of single machines. The fundamentals of these techniques are due to the work of Hansen and Salamon [Hansen and Salamon, 1990], who demonstrated that NNs performances could be improved significantly by means of combining some of them. In the literature, several methods for creating machine ensembles are proposed, see [Sharkey, 1999, Kuncheva, 2004, Rokach, 2010, Schapire and Freund, 2012].

Machine ensembles can be classified in different ways, as stated in [Sharkey, 1999, Haykin, 2007]. For example, they can be classified according to the way the learners collaborate to solve the problem. Using this criterion, we can divide machine ensembles in the following two families:

- Committees: Where all learners solve the same problem independently and, then, combine their outputs following some adequate criterion. Resampling techniques, such as Bootstrap [Sharkey, 1999] or Bagging (Bootstrap AGGregatING) [Breiman, 1996], along with other methods as Random Forest [Breiman, 2001] or Stacking [Wolpert, 1992], are included in this category.

- Consortia: Where the LMs in the ensemble cooperate among them. There are two possible approaches to form this kind of ensembles:
  - Using a divide and conquer strategy, i.e., decompose the problem in a set of subproblems that are assigned to different machines (or experts). This is the case of Mixtures of Experts described in [Jacobs et al., 1991, Jordan and Jacobs, 1994].
  - Adding machines to the ensemble sequentially, so that the performance of the overall machine is improved at each step, as it is the case of Boosting [Freund, 1995, Freund and Schapire, 1996].

It is important to note that these two philosophies are mutually exclusive, so that it is possible to combine them [Kuncheva, 2004].

1.2 Thesis motivation and objectives

GPs have shown to be very competitive for nonlinear regression tasks and can be considered one of the state-of-the-art methods. They are resilient to overfitting problems due to the (typically) reduced number of hyperparameters to be learned, which can be tuned easily using a continuous optimization of the evidence, and produce probabilistic predictions in a natural way. However, their $\mathcal{O}(N^3)$ scalability limits the scope of application of full GPs (i.e., not considering sparse approximations) to data sets with a few thousand training samples.

GPs covariance functions usually assume stationarity: The covariance between two outputs is modeled as a function of the difference of their corresponding inputs. Although this stationarity assumption allows an intuitive interpretation and allows to construct easily the covariance matrix via Bochner Theorem [Gibbs, 1997], it can be unrealistic for real-world problems. It is possible to use nonstationary covariance

functions, but they require prior knowledge of the nonstationarities, which excludes practical applications where this knowledge is usually not available.

On the other hand, as it is also the case of other traditional regression methods such as NNs or SVR, GPs also assumes stationary (homoscedastic) noise, i.e., the noise is considered constant throughout the input space. This can also limit the performance of GPs for many real-world applications, producing misleading predictions.

Although there exist GP models for heteroscedastic regression [Goldberg et al., 1998] and for nonstationary regression [Adams and Stegle, 2008], there are no model capable to describe nonstationarities in the unknown function and in the noise at the same time.

The main objective of this Thesis is to propose a GP model to achieve nonstationary and heteroscedastic regression.

It is difficult to propose a GP model (apart from the standard case) whose posterior, evidence, and predictive distribution can be inferred analytically; consequently, one desired property for the likelihoods of the GP models is log-concavity. This requirement allows the posterior distribution to be unimodal, which favors the convergence of approximate inference algorithms, avoiding to use additional tricks to force them to converge, as it is the case of the nonstationary models in [Goldberg et al., 1998] and [Adams and Stegle, 2008]. Then, a desired requisite for the nonstationary GP model objective of this Thesis is to have a log-concave likelihood.

Volatility forecasting in financial time series is a direct application of heteroscedastic regression methods. A GP method to predict volatility was introduced in [Girolami and Calderhead, 2011], using an efficient MCMC technique, called Riemann Manifold Hamiltonian Monte Carlo (RHMC), to sample from the analytically intractable posterior distribution of the model. The experimental re-

sults of RMHMC and the nonstandard variational approximation proposed in [Lázaro-Gredilla and Titsias, 2011] to make inference on the same model, show a better performance of both GP methods with respect to the Generalized AutoRegressive Conditional Heteroscedastic (GARCH) models [Bollerslev, 1986], commonly used to predict volatility. However, these GP methods seem to have gone unnoticed in the Econometrics community.

Although the variational approximation in [Lázaro-Gredilla and Titsias, 2011] provides accurate volatility predictions at a reduced cost (compared to RMHMC), a secondary objective of this Thesis is to develop a faster GP method to perform inference on this GP volatility model without losing prediction performance, which can be of practical interest for financial purposes.

1.3 Structure of the Thesis

The rest of the Thesis is organized as follows: In Chapter 2 we review the standard GP Regression (GPR), approximate inference methods, and MCMC techniques, and include an overview of GP heteroscedastic regression methods.

In Chapter 3 we present the Divisive GP (DGP) model to achieve nonstationary regression, including heteroscedastic noise cases. Since the model is not analytically tractable, we propose an Expectation Propagation (EP) formulation to approximate the posterior and a Elliptical Slice Sampling (ESS) implementation to sample from the exact posterior.

In Chapter 4 we introduce a Laplace approximation to make inference on the DGP model in order to reduce the computational burden with respect to the corresponding EP approximation.

Similarly, in Chapter 5 we propose to use the Laplace method to per-

form inference on the GP model for volatility forecasting described in [Girolami and Calderhead, 2011], offering accurate approximations to the posterior at a reduced computational cost, showing a very competitive performance with respect to other methods used to predict volatility.

Finally, in Chapter 6 we summarize the main contributions and results obtained, and we present the most relevant conclusions extracted from the work developed in this Thesis, discussing possible direct extensions of this work and other related research avenues.

Chapter 2

Gaussian Processes

The fundamentals of GPs are not a recent topic, specially for time series prediction. The basic theory goes back to the works of [Kolmogorov, 1939] and [Wiener, 1949] in the 1940's. Then, in the 1970's, GPs were applied to perform multivariate regression (for low dimensional input spaces) in the field of Geostatistics under the name of Kriging[1] [Matheron, 1973, Journel and Huijbregts, 1978]. In [O'Hagan and Kingman, 1978], GPs for regression were first introduced in the Statistics community, noticing that GPs could be applied in a general regression context. Finally, in [Williams and Rasmussen, 1996], GPs for regression (GPR) were presented to the Machine Learning community. A more exhaustive review in GPs can be found in [Rasmussen and Williams, 2006].

The rest of the chapter is orgainzed as follows. In Section 2.1 we review the definition of GP and related basic concepts. GP regression is described in Section 2.2. In Section 2.3 we present two approximate inference algorithms: Expectation Propagation (EP) and the Laplace method. Then, in Section 2.4 we review two

[1]The name is due to the South African mining engineer Daniel G. Krige.

Markov Chain Monte Carlo (MCMC) techniques: Metropolis-Hastings and Elliptical Slice Sampling (ESS). A review on GP heteroscedastic and nonstationary regression methods appears in Section 2.5. Finally, a brief summary in Section 2.6 closes the chapter.

2.1 Concept of GP

According to [Rasmussen and Williams, 2006] we can define a GP as a collection of random variables, any finite number of which have a joint Gaussian distribution.

A GP is fully specified by its mean function and its covariance function, which are defined as

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}\left[f(\mathbf{x})\right] \\ k(\mathbf{x},\mathbf{x}') &= \mathbb{E}\left[(f(\mathbf{x})-m(\mathbf{x}))(f(\mathbf{x}')-m(\mathbf{x}'))\right] = \mathrm{cov}(f(\mathbf{x}),f(\mathbf{x}')) \end{aligned} \tag{2.1}$$

where $f(\mathbf{x})$ is a real stochastic process with $\mathbb{R}^D \to \mathbb{R}$.. The GP can be represented as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x},\mathbf{x}')) \tag{2.2}$$

Usually, the mean function is assumed to be zero[2], although a priori knowledge about tendencies of the mean can be incorporated to the model in case necessary.

In this case the random variables represent the value of the function $f(\mathbf{x})$ at point $\mathbf{x} \in \mathbb{R}^D$. Then, the joint distribution of a set of random variables $\mathbf{f} = \{f(\mathbf{x}_n)\}_{n=1}^N$ is given by

$$p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{m}, K) \tag{2.3}$$

where $\mathbf{m} = [m(\mathbf{x}_1), ..., m(\mathbf{x}_1)]^T$ and $[K]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

[2]This can be achieved subtracting the sample mean from the data, and then, assuming a zero-mean model.

The definition of the covariance function yields K symmetric and ensures that the GP is consistent, i.e., it fulfills the marginalization property. This property implies that if, for example, the GP specifies $(\mathbf{f}_a, \mathbf{f}_b) \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, then it must also specify $\mathbf{f}_a \sim \mathcal{N}(\boldsymbol{\mu}_a, \Sigma_{aa})$ where Σ_{aa} is the relevant submatrix of Σ (see Appendix A).

The covariance function expresses our beliefs in the smoothness and the behavior of the underlying GP. According to (2.1), the covariance function provides an estimate of the correlation between two points. The sufficient and necessary condition for any function to be a covariance function is it being positive definite. This constraint implies that the covariance function is a valid Mercer kernel [Schölkopf and Smola, 2001], ensuring the covariance matrix K to be positive definite.

Usually, to allow more flexibility, additional parameters can be introduced into the covariance function. These parameters are referred as hyperparameters, as they do not parameterize the model itself, but its prior statistics. Hereafter, we will use the notation $k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta})$ to indicate the dependence of the covariance function on the set of hyperparameters $\boldsymbol{\theta}$.

2.2 GP for regression

In a regression task, we consider N input-output pairs $\mathcal{D} = \{\mathbf{x}_n \in \mathbb{R}^D, y_n \in \mathbb{R}\}_{n=1}^N$ which can be modeled as some unknown latent function $f(\mathbf{x})$ plus an independent noise

$$y(\mathbf{x}_n) = f(\mathbf{x}_n) + \varepsilon_n \tag{2.4}$$

GPR models, under a Bayesian approach, the underlying noiseless latent function $f : \mathbb{R}^D \to \mathbb{R}$ and the independent noise term ε placing a GP prior distribution with

a zero-mean function and an arbitrary covariance function $k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta})$, i.e.,

$$f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta})) \tag{2.5}$$

As mentioned before, the covariance function expresses the beliefs about the smoothness of the latent function f specifying the way two samples are correlated. Then, the values of $k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta})$ depend only on the inputs $\mathbf{x}, \mathbf{x}'$ (or in the difference of the inputs if the covariance function is stationary), and the set of hyperparameters given in $\boldsymbol{\theta}$.

Taking $\mathbf{f} = [f_1, ..., f_N]^T$ as the evaluation of f at the inputs $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$, we have a multivariate Gaussian prior given by

$$p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, K) \tag{2.6}$$

where $[K]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$.

For the noise term ε, a zero-mean and σ^2-variance Gaussian distribution is assumed, as it is the case of many other machine learning regression methods that try to minimize the mean squared error (MSE) on the training samples. It is straightforward to prove that minimization of the MSE corresponds to maximize a Gaussian model for the likelihood assuming stationary noise (and neglecting the noise term). In the case of GPR, the stationary Gaussian noise assumption leads to a Gaussian likelihood on the latent function f which can be expressed as

$$p(\mathbf{y}|\mathbf{f}, \mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2 I) \tag{2.7}$$

where $\mathbf{y} = [y_1, ..., y_N]^T$, I is the identity matrix (of dimension N), and σ^2 is the noise power hyperparameter. For the sake of clarity we consider this hyperparameter as a part of the set of hyperparameters given in $\boldsymbol{\theta}$[3].

[3]Then, $\boldsymbol{\theta}$ groups the hyperparameters of the covariance function and the likelihood of the model.

To obtain the posterior distribution on $\mathbf{f}$ we apply the Bayes theorem

$$p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \frac{p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})p(\mathbf{y}|\mathbf{f}, \mathbf{X}, \boldsymbol{\theta})}{p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})} \tag{2.8}$$

where $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$ is known as the evidence or the marginal likelihood and can be calculated integrating out f on the product of the GP prior and the likelihood (see Appendix A.3), resulting in a Gaussian distribution

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = \int p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})\ p(\mathbf{y}|\mathbf{f}, \mathbf{X}, \boldsymbol{\theta})\ d\mathbf{f} = \mathcal{N}(\mathbf{y}|\mathbf{0}, K + \sigma^2 I) \tag{2.9}$$

Then, since the prior, the likelihood, and the evidence are Gaussian, the posterior distribution of f at the training points will also be Gaussian (see Appendix A.3)

$$p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{f}|\boldsymbol{\mu}_\mathbf{f}, \Sigma_\mathbf{f}) \tag{2.10}$$

with

$$\begin{aligned} \boldsymbol{\mu}_\mathbf{f} &= K(K + \sigma^2 I)^{-1}\mathbf{y} \\ \Sigma_\mathbf{f} &= (K^{-1} + \sigma^{-2} I)^{-1} \end{aligned} \tag{2.11}$$

To adjust the set of hyperparameters $\boldsymbol{\theta}$, a procedure known as Maximum Likelihood of level II (ML-II) is applied. Under this approach, we search for a set of hyperparameters that maximizes the evidence in (2.9). To make the calculations simpler and more stable an equivalent formulation is used, consisting on seeking a (possibly local) maximum of the log-evidence given by

$$\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^T(K + \sigma^2 I)^{-1}\mathbf{y} - \frac{1}{2}\log|K + \sigma^2 I| - \frac{N}{2}\log(2\pi) \tag{2.12}$$

To provide a fully Bayesian model, we should place a prior on the set of hyperparameters and, then, integrate them out. However, this approach lead to analytically intractable integrals, that need from other computationally more expensive approximations to be solved. For this reason, this option is usually avoided.

To make predictions for unseen (test) samples, $\mathbf{x}_*$, the inferred predictive distribution is also Gaussian and can be calculated as

$$p(y_*|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \int p(f_*|\mathbf{f}, \mathbf{X}, \mathbf{y}, \boldsymbol{\theta})\; p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta})\; d\mathbf{f} = \mathcal{N}(y_*|\mu_*, \sigma_*^2) \tag{2.13}$$

with

$$\mu_* = \mathbf{k}_{*N}^T \left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{y} \tag{2.14}$$

$$\sigma_*^2 = k_{**} - \mathbf{k}_{*N}^T \left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{k}_{*N} + \sigma^2 \tag{2.15}$$

where $[\mathbf{k}_{*N}]_j = k(\mathbf{x}_*, \mathbf{x}_j; \boldsymbol{\theta})$, $k_{**} = k(\mathbf{x}_*, \mathbf{x}_*; \boldsymbol{\theta})$. As it can be observed, one of the advantages of GPs is that the estimates of the variance for test samples naturally appear in the model.

It can also be noticed the equivalence between the equation of the predictive mean in (2.14) and the corresponding equation of Wiener's predictor. As mentioned previously in this chapter, Wiener's research on stochastic processes for time domain can be considered the fundamental of GPs [Wiener, 1949].

Another interesting remark is that, although the variance estimates in (2.15) are input-dependent, the underlying GPR model is homoscedastic, i.e., it assumes that the noise is constant with power σ^2. This can be a restrictive assumption for many real applications, as argued in Chapter 1.

2.3 Approximate inference methods

In the standard GPR model described in the previous section all the calculations are analytically tractable. Unfortunately, for many models the calculations of the posterior, the evidence, and the predictive distribution are not analytically tractable. One of the possible solutions to sidestep this problem is to approximate the posterior with a Gaussian distribution. Then, this posterior approximation is used to calculate

the approximate evidence (needed to estimate the hyperparameters with the ML-II implementation) and the approximate predictive distribution.

Although several approaches have been proposed in the literature, we will focus on two of the most used approximate inference techniques: The Laplace method and EP, which have been used to make inference on the GP models proposed in Chapters 3-5.

2.3.1 Laplace approximation

The Laplace method can be used to find a Gaussian approximation[4] $q(\mathbf{f}|\mathbf{y})$ to the real posterior $p(\mathbf{f}|\mathbf{y})$ doing a second order Taylor expansion of $\log p(\mathbf{f}|\mathbf{y})$ around the mode of the posterior, so that

$$q(\mathbf{f}|\mathbf{y}) = \mathcal{N}(\mathbf{f}|\hat{\mathbf{f}}, A^{-1}) \propto \exp\left(-\frac{1}{2}(\mathbf{f}-\hat{\mathbf{f}})^T A(\mathbf{f}-\hat{\mathbf{f}})\right) \tag{2.16}$$

where

$$\hat{\mathbf{f}} = \arg\max_{\mathbf{f}} p(\mathbf{f}|\mathbf{y}) \tag{2.17}$$

is the maximum a posteriori (MAP) estimate of $\mathbf{f}$ and

$$A = -\nabla\nabla \log p(\mathbf{f}|\mathbf{y}) \tag{2.18}$$

is the negative Hessian of the log-posterior.

Assuming that the posterior is unimodal, the Laplace method aims to maximize the log-posterior given by

$$\begin{aligned} \log p(\mathbf{f}|\mathbf{y}) &= \log p(\mathbf{y}|\mathbf{f}) + \log p(\mathbf{f}) \\ &= \log p(\mathbf{y}|\mathbf{f}) - \frac{1}{2}\mathbf{f}^T K^{-1}\mathbf{f} - \frac{1}{2}\log|K| - \frac{N}{2}\log 2\pi \end{aligned} \tag{2.19}$$

[4]For the sake of clarity we will omit the conditioning on the input data $\mathbf{X}$ and the set of hyperparameters $\boldsymbol{\theta}$.

Differentiating (2.19) with respect to $\mathbf{f}$ we have

$$\nabla \log p(\mathbf{f}|\mathbf{y}) = \nabla \log p(\mathbf{y}|\mathbf{f}) - K^{-1}\mathbf{f} \tag{2.20}$$

$$\nabla\nabla \log p(\mathbf{f}|\mathbf{y}) = \nabla\nabla \log p(\mathbf{y}|\mathbf{f}) - K^{-1} = -W - K^{-1} \tag{2.21}$$

where $W = -\nabla\nabla \log p(\mathbf{y}|\mathbf{f})$ is the negative Hessian of the log-likelihood.

Then, to find the mode of the posterior we have to solve $\nabla \log p(\mathbf{f}|\mathbf{y}) = 0$, which is a self-consistent equation that can be solved, for example, applying the Newton method. In this case, the update equation for $\mathbf{f}$ will be given by

$$\begin{aligned} \mathbf{f}^{new} &= \mathbf{f} - (\nabla\nabla \log p(\mathbf{f}|\mathbf{y}))^{-1} \nabla \log p(\mathbf{f}|\mathbf{y}) \\ &= \mathbf{f} + (K^{-1} + W)^{-1}(\nabla \log p(\mathbf{y}|\mathbf{f}) - K^{-1}\mathbf{f}) \\ &= (K^{-1} + W)^{-1}(W\mathbf{f} + \nabla \log p(\mathbf{y}|\mathbf{f})) \end{aligned} \tag{2.22}$$

The mode of the true posterior will be unique if the likelihood of the model is log-concave with respect to the latent function f. In this case, the Laplace method converges to this unique maximum. In other case, the posterior will be multimodal, so that the unimodal approximation provided by the Laplace method can be inappropriate. Another limitation is the locality of the Laplace method, since the whole approximation is determined by local properties around the maximum of the true posterior. Moreover, the quality of the posterior approximation can be also limited if the mean and the mode of the true posterior are not close[5]. However, in cases were the likelihood is log-concave and the true posterior is quite Gaussian, the Laplace approximation becomes a simple and efficient algorithm to provide accurate posterior approximations.

[5]This is the case of the Laplace approximation applied to GP classification.

2.3.2 EP

EP [Minka, 2001] is a general approximation algorithm that is not only used for GPs, but also to a wide range of applications in Bayesian modeling. EP finds a Gaussian posterior approximation $q(\mathbf{f}|\mathbf{y})$ to the true posterior $p(\mathbf{f}|\mathbf{y})$ by minimizing the Kullback-Leibler (KL) divergence between two approximate marginal distributions, as we will explain next.

We can write the posterior distribution in (2.8) factorizing the likelihood over the training cases

$$p(\mathbf{f}|\mathbf{y}) = \frac{p(\mathbf{f}) \prod_{n=1}^{N} p(y_n|f_n)}{p(\mathbf{y})} \tag{2.23}$$

but in this case, the local likelihoods $p(y_n|f_n)$ are not Gaussian, as was the case for (2.8), leading to an intractable posterior.

Under the EP algorithm, we define unnormalized Gaussian approximations to the local likelihood terms. These approximations are known as site functions. They can be described as

$$p(y_n|f_n) \simeq t(f_n|\tilde{Z}_n, \tilde{\mu}_n, \tilde{\sigma}_n^2) = \tilde{Z}_n \mathcal{N}(f_n|\tilde{\mu}_n, \tilde{\sigma}_n) \tag{2.24}$$

where $\tilde{Z}_n$, $\tilde{\mu}_n$, and $\tilde{\sigma}_n^2$ are defined as the site parameters. Then, the product of the site functions on the training cases can be also expressed as an unnormalized multivariate distribution given by

$$\prod_{n=1}^{N} t(f_n|\tilde{Z}_n, \tilde{\mu}_n, \tilde{\sigma}_n^2) = \mathcal{N}(\mathbf{f}|\tilde{\boldsymbol{\mu}}, \tilde{\Sigma}) \prod_{n=1}^{N} \tilde{Z}_n \tag{2.25}$$

where $\tilde{\boldsymbol{\mu}}$ is the vector of $\tilde{\mu}_n$ and $\tilde{\Sigma}$ is a diagonal matrix with $[\tilde{\Sigma}]_{nn} = \tilde{\sigma}_n^2$. Then, we can write the expression of the approximate posterior as

$$q(\mathbf{f}|\mathbf{y}) = \frac{1}{Z_{\text{EP}}}\, p(\mathbf{f}) \prod_{n=1}^{N} t(f_n|\tilde{Z}_n, \tilde{\mu}_n, \tilde{\sigma}_n^2) = \mathcal{N}(\mathbf{f}|\boldsymbol{\mu}, \Sigma) \tag{2.26}$$

with

$$\begin{aligned} \boldsymbol{\mu} &= \Sigma\tilde{\Sigma}^{-1}\tilde{\boldsymbol{\mu}} \\ \Sigma &= (K^{-1} + \tilde{\Sigma}^{-1})^{-1} \end{aligned} \tag{2.27}$$

and Z_{EP} is the normalizing constant which corresponds to the approximate marginal likelihood $q(\mathbf{y})$.

To select the site functions parameters the minimization of $\mathrm{KL}\left[p(\mathbf{f}|\mathbf{y})||q(\mathbf{f}|\mathbf{y})\right]$, the KL divergence between the true and the approximate posterior, would be appealing. However, the direct minimization of this KL divergence is intractable. Instead, EP sequentially updates individual site functions, minimizing the following KL divergence

$$\mathrm{KL}\left[q_{\backslash n}(f_n)p(y_n|f_n)||q_{\backslash n}(f_n)t(f_n|\tilde{Z}_n, \tilde{\mu}_n, \tilde{\sigma}_n^2)\right] \tag{2.28}$$

where $q_{\backslash n}(f_n)$ is known as the cavity distribution [Opper and Winther, 2000], which is a Gaussian distribution given by the integral of the product of the GP prior and all the site functions, except the site $t(f_n)$ whose moments are going to be estimated:

$$q_{\backslash n}(f_n) = \mathcal{N}(f_{\backslash n}|\mu_{\backslash n}, \sigma_{\backslash n}^2) \propto \int p(\mathbf{f}) \prod_{j \neq n} t(f_j|\tilde{Z}_j, \tilde{\mu}_j, \tilde{\sigma}_j^2)\, df_j \tag{2.29}$$

with

$$\begin{aligned} \sigma_{\backslash n}^2 &= (\sigma_n^{-2} + \tilde{\sigma}_n^{-2})^{-1} \\ \mu_{\backslash n} &= \sigma_{\backslash n}^{-2}(\sigma_n^{-2}\mu_n + \tilde{\sigma}_n^{-2}\tilde{\mu}_n) \end{aligned} \tag{2.30}$$

where μ_n is the n-th element of the vector $\boldsymbol{\mu}$ and $\sigma_n^2 = [\Sigma]_{nn}$.

As the cavity function is Gaussian, the right side of the KL divergence in (2.28) is an unnormalized Gaussian. Then, it is well known that, provided a distribution $q(\mathbf{x})$ is Gaussian, the distribution $q(\mathbf{x})$ that minimizes $KL(p(\mathbf{x})||q(\mathbf{x}))$ (where no restrictions are imposed to distribution $p(\mathbf{x})$) is the one whose first and second order moments match. In this case, we also need the zeroth moment matching to include the normalization factor on the right term of (2.28). Defining

$$\hat{q}(f_n) = q_{\backslash n}(f_n)t(f_n|\tilde{Z}_n, \tilde{\mu}_n, \tilde{\sigma}_n^2) = \hat{Z}_n\mathcal{N}(f_n|\hat{\mu}_n, \hat{\sigma}_n^2) \tag{2.31}$$

the zeroth, first and second order moments of $\hat{q}(f_n)$ that minimize the KL divergence in (2.28) can be calculated as

$$\hat{Z}_n = \int q_{\backslash n}(f_n)p(y_n|f_n)\ df_n \tag{2.32}$$

$$\hat{\mu}_n = \int f_n\ q_{\backslash n}(f_n)p(y_n|f_n)\ df_n \tag{2.33}$$

$$\hat{\sigma}_n^2 = \int (f_n - \hat{\mu}_n)^2\ q_{\backslash n}(f_n)p(y_n|f_n)\ df_n \tag{2.34}$$

Finally, using equations (A.5) and (A.6), the site parameters are calculated as

$$\tilde{\sigma}_n^2 = (\hat{\sigma}_n^2 - \sigma_{\backslash n}^2)^{-1} \tag{2.35}$$

$$\tilde{\mu}_n = \tilde{\sigma}_n^2(\hat{\sigma}_n^2\ \hat{\mu}_n - \sigma_{\backslash n}^2\ \mu_{\backslash n}) \tag{2.36}$$

$$\tilde{Z}_n = \hat{Z}_n\ \sqrt{2\pi(\sigma_{\backslash n}^2 + \tilde{\sigma}_n^2)}\exp\left(\frac{(\mu_{\backslash n} - \tilde{\mu}_n)^2}{2(\sigma_{\backslash n}^2 + \tilde{\sigma}_n^2)}\right) \tag{2.37}$$

To obtain the EP approximation to the marginal likelihood, Z_{EP} in (2.26) we need to compute

$$Z_{\mathrm{EP}} = q(\mathbf{y}) = \int p(\mathbf{f})\ \prod_{n=1}^{N} t(f_n|\tilde{Z}_n, \tilde{\mu}_n, \tilde{\sigma}_n^2)\ d\mathbf{f} \tag{2.38}$$

2.4 MCMC methods

Another way to perform inference on intractable GP models is the use of MCMC techniques to sample from the exact posterior. These methods are known to be asymptotically exact, in contrast to the approximate inference methods, however, the computational cost to obtain these asymptotically unbiased estimates of the posterior is extremely high.

In the next subsections we will briefly review two methods to perform MCMC inference on GPs: Metropolis-Hastings [Hastings, 1970] and Elliptical Slice Sampling

(ESS) [Murray et al., 2010]. The latter technique is used to perform inference on the divisive GP model in Chapters 3 and 4.

2.4.1 Metropolis-Hastings

The objective of MCMC techniques applied to GPs is to sample from a posterior distribution at the training points which is proportional to the product of the GP prior and the likelihood. In the Metropolis-Hastings method for GPs introduced by [Neal, 1999], given an initial state $\mathbf{f}_t$, which corresponds to a sample drawn from the posterior of $\mathbf{f}$, and a step-size parameter $\eta \in [-1, 1]$, a new state $\mathbf{f}_{t+1}$ is proposed according to

$$\mathbf{f}_{t+1} = \sqrt{1-\eta^2}\ \mathbf{f}_t + \eta\ \boldsymbol{\nu} \tag{2.39}$$

where $\boldsymbol{\nu}$ is a sample from the GP prior, so that

$$p(\boldsymbol{\nu}) = \mathcal{N}(\boldsymbol{\nu}|\mathbf{0}, K) \tag{2.40}$$

The proposed new state is accepted with probability

$$p(\text{accept}) = \min\left(1, \frac{p(\mathbf{y}|\mathbf{f}_{t+1})}{p(\mathbf{y}|\mathbf{f}_t)}\right) \tag{2.41}$$

This method is simple to implement and can immediately be applied to many GP models. However, as reported by [Neal, 1999], η needs to be chosen appropriately for the Markov chain to reach the stationary state in a reasonable amount of iterations, i.e., to mix efficiently. This usually requires to run several chains with different η to chose a proper value for the step-size parameter.

2.4.2 ESS

To automatically search over the step-size parameter η, ESS proposes a reparameterization of the update equation (2.39). Given $\mathbf{f}_t$ and $\boldsymbol{\nu}$ in (2.39), and η varying in

the interval $[-1, 1]$, we have the expression of half of an ellipse, with loci $\mathbf{f}_t$ and $\boldsymbol{\nu}$. Instead, ESS proposes a more natural parameterization

$$\mathbf{f}_{t+1} = \nu \ \sin\alpha + \mathbf{f}_t \ \cos\alpha \tag{2.42}$$

which defines a full ellipse. Then, for a given α there is an equivalent η that gives the same proposal. However, the full ellipse gives a richer choice to search over the step-size. The ESS algorithm is shown in Table 2.1. It uses Slice Sampling (SS) [Neal, 2003] to sample along the possible values of α, reducing the search interval for this parameter when the new proposals $\mathbf{f}_{t+1}$ are rejected.

Table 2.1: Pseudocode of the ESS algorithm

Input: current state $\mathbf{f}_t$
Output: new state $\mathbf{f}_{t+1}$

1 - Draw $\boldsymbol{\nu} \sim \mathcal{N}(\boldsymbol{\nu}, \mathbf{0}, K)$
2 - Set the likelihood threshold p^*:
  $u \sim \text{Uniform}[0, 1]$
  $\log p^* \leftarrow \log p(\mathbf{y}|\mathbf{f}_t) + \log u$
3 - Draw an initial proposal (also defining a bracket for α):
  $\alpha \sim \text{Uniform}[0, 2\pi]$
  $[\alpha_{\min}, \alpha_{\max}] \leftarrow [\alpha - 2\pi, \alpha]$
4 - $\mathbf{f}_{t+1} \leftarrow \mathbf{f}_t \ \cos\alpha + \boldsymbol{\nu} \ \sin\alpha$
5 - if $p(\mathbf{y}|\mathbf{f}_{t+1}) > \log p^*$ then:
  Accept proposal and return $\mathbf{f}_{t+1}$
6 - else:
  Shrink the bracket and try a new point:
    if $\alpha < 0$ then:
      $\alpha_{\min} \leftarrow \alpha$
    else:
      $\alpha_{\max} \leftarrow \alpha$
    $\alpha \sim \text{Uniform}[\alpha_{\min}, \alpha_{\max}]$
    Go to 4

2.5 Nonstationary and heteroscedastic GPR methods

The standard GP model assumes stationarity, i.e., the noise power remains constant throughout input space (homoscedasticity) and the covariance between two observations is typically modeled as dependent only on the difference between corresponding inputs. Though this stationarity assumption does yield useful and analytically tractable models, it is often not honored by real-world data sets.

Trying to overcome this limitation, non-parametric heteroscedastic models with input-dependent noise power have been developed. Not being analytically tractable as standard GPs are, different approximate inference techniques have been employed to model the mean and the log-noise latent functions, so that the likelihood for a given point is:

$$p(y_n|f_n, g_n) = \mathcal{N}\left(y_n|f_n, e^{g_n}\right) \tag{2.43}$$

To make inference on this model, Gibbs sampling is used in [Goldberg et al., 1998] to sample from the exact posterior on both latent functions. However, the method lacks a search procedure to maximize the evidence with respect to the hyperparameters. In [Kersting et al., 2007], a point estimation of the log-noise is performed using an iterative algorithm. Nevertheless, the method is not guaranteed to converge and may oscillate. This issue is addressed in [Quadrianto et al., 2009] by choosing the log-noise so as to maximize a penalized likelihood, equivalent up to a constant to the prior on the log-noise. A non-standard variational approximation and EP posterior approximations are proposed in [Lázaro-Gredilla and Titsias, 2011] and [Muñoz-González et al., 2011] respectively. However, the likelihood in (2.43) used by all these methods is not log-concave, so that the posterior is not log-concave, thus, not ensuring a unimodal posterior.

In contrast, [Le et al., 2005] proposes a GP model using natural parameters to ensure posterior log-concavity, although inference is also not analytically tractable.

They apply the Newton method to solve a convex optimization problem that provides MAP estimates of the mean and the input-dependent noise.

A more flexible approach is presented in [Adams and Stegle, 2008], where observations are purported as the product of realizations of two independent GPs, and EP is used for approximate inference. This allows to model low-variation amplitude nonstationarities, although the proposed model does not include the heteroscedastic noise case explicitly, as we will describe next.

In the following subsections we describe the two most prominent algorithms in the literature for GP heteroscedastic and nonstationary regression: the Variational Heteroscedastic GPR (VHGPR) [Lázaro-Gredilla and Titsias, 2011] and the GP Product Model (GPPM) [Adams and Stegle, 2008].

2.5.1 VHGPR

This method uses a non-standard variational approximation to perform inference in the heteroscedastic form given in (2.43). Although the evidence of this model cannot be computed analytically, it is possible to lower bound it variationally with the following analytically tractable expression

$$F(q(\mathbf{f}), q(\mathbf{g})) = \log p(\mathbf{y}) - \mathrm{KL}\left(q(\mathbf{f})q(\mathbf{g}) || p(\mathbf{f}, \mathbf{g}|\mathbf{y})\right) \tag{2.44}$$

where $q(\mathbf{f})$ and $q(\mathbf{g})$ are variational probability densities and $p(\mathbf{f}, \mathbf{g}|\mathbf{y})$ is the true posterior. It is clear that $F(q(\mathbf{f}), q(\mathbf{g}))$ is a lower bound of the evidence, as $p(\mathbf{y})$ is independent of the variational densities and the KL divergence is a positive value.

The lower bound (2.44) depends on two N-dimensional variational distributions, but it is possible to obtain a simpler and tighter bound, called the Marginalized Variational (MV) bound, by removing the dependence on one of the variational distributions. Then, in this case, to optimally remove the dependence of (2.44) on

$q(\mathbf{f})$, we need to compute the distribution $q^*(\mathbf{f})$ that maximizes (2.44) and inserting it back into the bound.

Then, the MV bound reduces the number of variational parameters and provides a tighter lower bound to the evidence, as shown in [Lázaro-Gredilla and Titsias, 2011]. Moreover, the proposed formulation makes the variational algorithm converge despite the likelihood (2.43) is not log-concave.

The experimental results show that the posterior approximations of VHGPR are accurate when compared to an MCMC implementation with ESS for the same model.

2.5.2 GPPM

GPPM proposes a nonstationary regression model which consists on the pointwise product of two latent functions, plus independent Gaussian noise. The corresponding likelihood is:

$$p(y_n|f_n, g_n) = \mathcal{N}(y_n|f_n e^{g_n}, \sigma^2) \tag{2.45}$$

where σ^2 is the (homoscedastic) noise hyperparameter. The objective of the product of the two latent functions is that f captures local near-stationary variations and g models slowly-varying amplitude nonstationarities.

Since the likelihood is not jointly Gaussian on both latent functions, it is not possible to make inference in the model analytically. Then, the authors propose to use EP to approximate the posterior distribution. One of the drawbacks of the model is that the likelihood is not log-concave, so that the posterior is not unimodal, which hinders the convergence of EP. To alleviate this problem, the authors use two techniques: Skipping and damping [Minka, 2001]. The first one consists on avoiding site functions updates when the update equation of the variance (2.35) leads to negative (and invalid) values of variance. On the other hand, damping the site functions update equations (2.35) - (2.37) favors the EP approximation to converge

smoothly. For example, to update the mean of a given site function $t(f_n)$ using damping we have:

$$\tilde{\mu}_n = (1 - \lambda)\ \tilde{\mu}_n^{\text{old}} + \lambda\ \tilde{\mu}_n^{\text{new}} \tag{2.46}$$

where $\tilde{\mu}_n^{\text{new}} = \tilde{\sigma}_n^2(\hat{\sigma}_n^2\ \hat{\mu}_n - \sigma_{\setminus n}^2\ \mu_{\setminus n})$ and $0 < \lambda < 1$ is the damping factor.

Another limitation of GPPM is that the calculations of the moments in (2.32) - (2.34), needed to update the parameters of the site functions, are not analytically tractable. Then, the authors propose to use the approach in [Zoeter and Heskes, 2005], where the Gauss-Hermite approximation is applied to calculate the integrals for these moments. The authors argue that this approximation causes that the approximate evidence and its derivatives with respect to the hyperparameters are numerically not stable to provide precise gradients. Then, it is not possible to apply an ML-II implementation to search the hyperparameters.

Finally, as we have mentioned before, GPPM only models amplitude nonstationarities and does not assume noise heteroscedascity, limiting the capabilities of the model.

2.6 Summary

In this chapter we have reviewed GPs, a powerful nonparametric framework for nonlinear regression. GPs place a prior probability density over the unknown function. Then, proceeding in a purely Bayesian fashion, the prior over functions and the likelihood implied by the assumed observation model can be used to infer the posterior distribution over the sought unknown function given data. GPs offer a number of advantages with respect to traditional regression machines, such as to naturally and easily produce probabilistic predictions, i.e., average and dispersion values, and resistance to overfitting due to their use of a reduced number of hyperparameters, that can be tuned with a simple continuous optimization of the evidence. However, the

main drawback of GPs is the $\mathcal{O}(N^3)$ cost of performing inference with the standard model.

The standard GPR regards observations as the sum of some unknown function of the inputs plus Gaussian noise, as it is common in most regression approaches. In this case, the posterior distribution over the unknown latent function, the evidence, and the predictive distribution for test samples, are Gaussian. For non-Gaussian likelihoods, these calculations are not analytically tractable. To make inference on those models we have two families of approaches. The first one consists on making a Gaussian approximation of the posterior, as it is the case of the Laplace method and EP. The second family includes MCMC techniques to sample from the exact posterior, as the Metropolis-Hastings or the ESS algorithms described in this chapter.

The stationarity of the unknown latent function and the Gaussian noise assumed by the standard GP model can be too restrictive for many real-world applications. To overcome these limitations, there are GP techniques that allow to model the log-power of input dependent noise along with the underlying mean function, as it is the case of VHGPR. On the other hand, the GPPM model allows to model amplitude nonstationarities, although it does not include heteroscedascity in the model.

Given the limitations of the revised GP methods to model nonstationarity, we propose a Divisive GP model that allows to model both amplitude nonstationarities and heteroscedastic noise, as we will describe in the next chapter.

Chapter 3

Divisive Gaussian Processes for nonstationary regression using Expectation Propagation

In order to overcome the limitations of the standard GPR to model nonstationary scenarios, we propose a Divisive GP (DGP) model where the pointwise division of two stationary latent functions allows dealing with amplitude nonstationarities and heteroscedastic noise for regression tasks.

The log-concavity of the likelihood favors the convergence of approximate inference methods, such as EP or the Laplace approximation, in contrast to other proposed approaches that we described in Chapter 2, such as GPPM [Adams and Stegle, 2008] or the heteroscedastic model used by [Goldberg et al., 1998, Kersting et al., 2007, Quadrianto et al., 2009, Muñoz-González et al., 2011, Lázaro-Gredilla and Titsias, 2011], where the proposed likelihoods are not log-concave, leading to convergence problems in some

cases. Moreover, the aforementioned models are not capable to model amplitude nonstationarities and heteroscedastic noise at the same time, as it is the case of the proposed DGP model.

This chapter is organized as follows: In Section 3.1 we present the DGP model under a Bayesian framework. Inference procedures on the DGP model with ESS and EP are described in Section 3.2. Experimental results on synthetic and real data sets are shown in Section 3.3. Finally, we summarize the chapter in Section 3.4.

3.1 The DGP model

The DGP model we propose in this chapter can be viewed as a possibly noisy stationary latent function f modulated by another stationary latent function g which describes amplitude nonstationarities affecting to both the latent function f and the noise associated to f. Thus, the observations under the DGP model are described as

$$y(\mathbf{x}_n) = \frac{f(\mathbf{x}_n)}{g^+(\mathbf{x}_n)} + \varepsilon_n \tag{3.1}$$

where $f(\mathbf{x})$ is the noisy stationary latent function, $g^+(\mathbf{x})$ is the modulating function defined as the positive part of some noise-free latent function $g(\mathbf{x})$, i.e., $g^+(\mathbf{x}) = \max(g(\mathbf{x}), 0)$, and ε_n is an input-dependent Gaussian noise term that can be modelled as $\varepsilon \sim \mathcal{N}(0, c/(g^+(\mathbf{x}))^2)$, where c is a noise power constant scale factor. The likelihood of the proposed model for $f(\mathbf{x})$ and $g(\mathbf{x})$ given an observation $y(\mathbf{x})$ can be written as

$$p(y_n|f_n, g_n) = \mathcal{N}(y_n|f_n/g_n^+, c/(g_n^+)^2) \tag{3.2}$$

where $f_n = f(\mathbf{x}_n)$, $g_n^+ = g^+(\mathbf{x}_n)$, and $y_n = y(\mathbf{x}_n)$.

Note that the DGP model includes the standard GPR model as a particular case if the latent function g is constant.

Following a Bayesian treatment, we place GP priors on f and g, so that

$$\begin{aligned} f(\mathbf{x}) &\sim \mathcal{GP}(0, k_f(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}_f) + \sigma_f^2 \, \delta_{\mathbf{x}\mathbf{x}'}) \\ g(\mathbf{x}) &\sim \mathcal{GP}(\mu_0, k_g(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}_g)) \end{aligned} \tag{3.3}$$

where $k_f(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}_f)$, $k_g(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}_g)$, are any valid covariance functions parameterized by $\boldsymbol{\theta}_f$ and $\boldsymbol{\theta}_g$; σ_f^2 is a homoscedastic noise hyperparameter, $\delta_{\mathbf{x}\mathbf{x}'}$ is the Kronecker delta, and c is a constant that, together with the mean μ_0, modulates the mean power of the heteroscedastic noise. Thus, the model is fully specified and depends only on the covariance functions hyperparameters $\boldsymbol{\theta}_f$, σ_f^2, $\boldsymbol{\theta}_g$, μ_0, and the noise term c.

It can be appreciated that the noise in the DGP model is described by two terms. First, the hyperparameter σ_f^2 represents a stationary noise term associated to the latent function f. On the other hand, $c/(g^+(\mathbf{x}))^2$ models noise nonstationarities. Having this in mind, it can be observed that the mean of the noise power can be specified in both σ_f^2 and c/μ_0^2. Note that this is equivalent to a model that neglects σ_f^2, i.e., considering f a noise-free latent function, assuming that all the mean noise power is in the term c/μ_0^2. However, this redundancy in the DPG model can favour hyperparameter learning.

The proposed model has some relevant differences with respect to GPPM [Adams and Stegle, 2008]. As described in Chapter 2, this model consists on a noise-free stationary latent function f multiplied by another stationary (modulating) latent function g that describes slow-variation amplitude nonstationarities. The likelihood of GPPM given in (2.45) contrasts the DGP likelihood in (3.2), where the function g modulates a noisy function, so the model includes heteroscedastic noise cases. Consequently, it can be applied to nonstationary heteroscedastic problems. GPPM also has an important drawback for achieving variational inference as the likelihood is not log-concave, which leads to a multimodal posterior. This caused convergence problems for the proposed EP approximation in [Adams and Stegle, 2008]. Contrarily, the likelihood of DGP model (3.2) is log-concave on both f and g. Thus, the posterior is log-concave and unimodal, which suits EP Gaussian posterior approximation and

favors convergence, although it is not guaranteed. Another remarkable limitation of GPPM is that the ML-II hyperparameters search cannot be performed. Instead, a grid search is proposed to set the hyperparameters. As the authors point out, the gradients they used for the ML-II implementation are approximate. This may be due to the loss of accuracy in the double Gauss-Hermite approximation needed to calculate the site functions moments in the EP algorithm. In the case of the DGP model, the values of the site functions moments can be analytically obtained. Therefore, it is possible to use a ML-II implementation to search the hyperparameters.

3.2 Inference

Given the DGP model in (3.1) and the likelihood and GP priors defined in (3.2) and (3.3), respectively, we can write the posterior over the latent functions f and g at the training points as

$$p(\mathbf{f}, \mathbf{g}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{f}, \mathbf{g})p(\mathbf{f})p(\mathbf{g})}{p(\mathbf{y})} \tag{3.4}$$

where $p(\mathbf{y})$ is the marginal likelihood or evidence, which can be expressed as

$$p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{f}, \mathbf{g})p(\mathbf{f})p(\mathbf{g})d\mathbf{f}d\mathbf{g} \tag{3.5}$$

For the sake of simplicity we omit the conditioning on the training inputs $\mathbf{X}$ and the set of hyperparameters $\boldsymbol{\theta} = [\boldsymbol{\theta}_f, \boldsymbol{\theta}_g, \mu_0, c]^T$.

The likelihood of the DGP model given in (3.2) is not Gaussian on both latent functions (although it is Gaussian on f), which makes analytical inference of the evidence intractable. To overcome this limitation, we propose to apply the EP algorithm to approximate the posterior distribution over f and g. To assess the quality of the EP approximation we also provide a MCMC implementation with ESS to draw samples from the exact posterior.

3.2.1 Exact inference with ESS

As explained in Chapter 2, MCMC techniques allow to sample from posterior distributions, specially for cases in which an analytically tractable solution is not possible. The main drawback of MCMC methods is the high computational burden that scales in time with $\mathcal{O}(N^3)$, which implies a very high cost even for moderate-size data sets.

We use ESS to draw samples from the posterior in (3.4). This method is specially suitable to sample from posteriors with tightly correlated Gaussian priors, as in this case. To sample from the DGP posterior, ESS allows to make joint updates on f and g. As described in Chapter 2, the algorithm does not have a step-size parameter as in the case of Metropolis-Hastings. Instead, an elliptical parametrization is proposed to to update the state of the sampler, using the slice sampling method presented in [Neal, 2003] with an adaptive step-size.

Then, as in (2.42), given an initial state $\boldsymbol{\phi}_t = [\mathbf{f}_t, \mathbf{g}_t]^T$ a new state $\boldsymbol{\phi}_{t+1}$ is proposed according to

$$\boldsymbol{\phi}_{t+1} = \boldsymbol{\nu} \sin\alpha + \boldsymbol{\phi}_t \cos\alpha \tag{3.6}$$

where $\boldsymbol{\nu} = [\boldsymbol{\nu}_f, \boldsymbol{\nu}_g]^T$ is drawn from the GP prior distribution

$$\boldsymbol{\nu} \sim p(\boldsymbol{\nu}_f)\; p(\boldsymbol{\nu}_g) = \mathcal{N}(\boldsymbol{\nu}_f|\mathbf{0}, K_f)\; \mathcal{N}(\boldsymbol{\nu}_g|\mu_0\mathbf{1}, K_g) \tag{3.7}$$

K_f and K_g are positive definite covariance matrices from the GP priors on f and g defined in (3.3). The value of α is initially sampled from a uniform distribution in the interval $[0, 2\pi]$. If the proposal (3.7) is rejected, then, new values of α are drawn, restricting the interval of the uniform distribution in the way showed in Table 2.1 to generate new proposals.

3.2.2 Approximate inference with EP

With EP, we can approximate the real posterior $p(\mathbf{f}, \mathbf{g}|\mathbf{y})$ in (3.4) with a multivariate Gaussian distribution, $q(\mathbf{f}, \mathbf{g}|\mathbf{y})$, which allows for analytical tractability. As explained in Chapter 2, the direct minimization of the KL divergence between the real and the approximate posterior $\mathrm{KL}[p(\mathbf{f}, \mathbf{g}|\mathbf{y})||q(\mathbf{f}, \mathbf{g}|\mathbf{y})]$ is not tractable. Instead, we can sequentially update individual approximations to the likelihood for each training sample.

EP uses an unnormalized Gaussian distribution, called the site function, to approximate the local likelihood. In this case, to approximate the local likelihood in (3.2) we use an unnormalized bivariate Gaussian distribution that can be expressed as

$$p(y_n|f_n, g_n) \simeq t_n(\boldsymbol{\phi}_n|\tilde{Z}_n, \tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n) \equiv \tilde{Z}_n \, \mathcal{N}(f_n|\tilde{\mu}_{f_n}, \tilde{\sigma}_{f_n}) \, \mathcal{N}(g_n|\tilde{\mu}_{g_n}, \tilde{\sigma}_{g_n}) \tag{3.8}$$

where $\boldsymbol{\phi}_n = [f_n, g_n]^T$, $\tilde{\boldsymbol{\mu}}_n = [\tilde{\mu}_{f_n}, \tilde{\mu}_{g_n}]^T$, and $\tilde{\boldsymbol{\Sigma}}_n$ is a covariance matrix with $\tilde{\sigma}_{f_n}$ and $\tilde{\sigma}_{g_n}$ in its main diagonal.

The likelihood $p(\mathbf{y}|\mathbf{f}, \mathbf{g})$ is approximated by the product of the N site functions, that can be arranged as

$$p(\mathbf{y}|\mathbf{f}, \mathbf{g}) \simeq \prod_{n=1}^{N} t_n = \mathcal{N}(\mathbf{f}|\tilde{\boldsymbol{\mu}}_f, \tilde{\Sigma}_f) \, \mathcal{N}(\mathbf{g}|\tilde{\boldsymbol{\mu}}_g, \tilde{\Sigma}_g) \prod_{n=1}^{N} \tilde{Z}_n \tag{3.9}$$

where $\tilde{\boldsymbol{\mu}}_f$ and $\tilde{\boldsymbol{\mu}}_g$ are N-length vectors with $\tilde{\mu}_{f_n}$ and $\tilde{\mu}_{g_n}$ at the n-th position, respectively. The covariances $\tilde{\Sigma}_f$ and $\tilde{\Sigma}_g$ are diagonal matrices of size N with $\tilde{\sigma}^2_{f_n}$ and $\tilde{\sigma}^2_{g_n}$ at the n-th position of the diagonal, respectively. Then, the expression of the approximate posterior is given by

$$q(\mathbf{f}, \mathbf{g}|\mathbf{y}) = \frac{1}{q(\mathbf{y})} p(\mathbf{f}) \, p(\mathbf{g}) \prod_{n=1}^{N} t_n = \frac{1}{q(\mathbf{y})} \mathcal{N}(\mathbf{f}|\mathbf{0}, K_f) \, \mathcal{N}(\mathbf{g}|\mu_0 \mathbf{1}, K_g) \prod_{n=1}^{N} t_n \tag{3.10}$$

where K_f and K_g are positive definite covariance matrices from the GP priors on f

and g defined in (3.3). This approximate posterior can also be expressed as

$$q(\mathbf{f}, \mathbf{g}|\mathbf{y}) = \mathcal{N}(\mathbf{f}|\boldsymbol{\mu}_f, \Sigma_f)\, \mathcal{N}(\mathbf{g}|\boldsymbol{\mu}_g, \Sigma_g) \tag{3.11}$$

with

$$\begin{aligned} \Sigma_f &= (K_f^{-1} + \tilde{\Sigma}_f^{-1})^{-1} \\ \boldsymbol{\mu}_f &= \Sigma_f \tilde{\Sigma}_f^{-1} \tilde{\boldsymbol{\mu}}_f \\ \Sigma_g &= (K_g^{-1} + \tilde{\Sigma}_g^{-1})^{-1} \\ \boldsymbol{\mu}_g &= \Sigma_g (\tilde{\Sigma}_g^{-1} \tilde{\boldsymbol{\mu}}_g + K_g^{-1} \mu_0 \mathbf{1}) \end{aligned} \tag{3.12}$$

The approximate evidence $q(\mathbf{y})$ given by EP can be written as

$$q(\mathbf{y}) = \int \mathcal{N}(\mathbf{f}|\mathbf{0}, K_f)\, \mathcal{N}(\mathbf{g}|\mu_0 \mathbf{1}, K_g) \prod_{n=1}^{N} t_n \, d\mathbf{f} \, d\mathbf{g} \tag{3.13}$$

The cavity distribution, needed to compute the moments of the site functions, corresponds to a bivariate Gaussian distribution that can be factorized as follows:

$$q_{\setminus n}(\phi_n) = q_{f_{\setminus n}}(f_n)\, q_{g_{\setminus n}}(g_n) = \mathcal{N}(f_n|\mu_{f_{\setminus n}}, \sigma^2_{f_{\setminus n}})\, \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma^2_{g_{\setminus n}}) \tag{3.14}$$

where the expressions to calculate the mean and the variance of the two Gaussians are given in (2.30).

As explained in Chapter 2, EP updates the moments of t_n iteratively by minimizing the KL divergence between the product of the cavity distribution and the real (non-Gaussian) local likelihood (3.2), and the product of the cavity distribution and the local site function t_n, as shown in (2.28). First and second order moment matching is needed to minimize this KL divergence. However, as t_n is an unnormalized Gaussian, the normalization parameter of the site function $\tilde{Z}_n$ is necessary to compute the approximate evidence (3.13). Therefore, we also need to include zeroth order moment matching of both distributions in the KL divergence.

Moment calculation

The next step involves the calculation of the zeroth, first and second order moments of the left distribution in the KL divergence (2.28). The details of the calculations are presented in Appendix C.1.

The zeroth order moment results

$$\hat{Z}_n = \iint q_{\backslash n}(\phi_n)\; p(y_n|f_n, g_n)\; df_n\; dg_n = \breve{Z}\; \breve{\mu}_{g_t}\; \Phi\left(-\frac{\breve{\mu}_g}{\breve{\sigma}_g}\right) \tag{3.15}$$

where

$$\breve{Z} = \mathcal{N}\left(\mu_{f_{\backslash n}} | \mu_{g_{\backslash n}} y_n, c + \sigma^2_{f_{\backslash n}} + \sigma^2_{g_{\backslash n}} y_n^2\right) \tag{3.16}$$

$\breve{\mu}_{g_t}$ is the mean of the Gaussian $\mathcal{N}(g_n|\breve{\mu}_g, \breve{\sigma}_g^2)$ truncated to the positive values of g_n with

$$\begin{aligned} \breve{\sigma}_g^2 &= \left(\sigma^{-2}_{g_{\backslash n}} + \frac{y_n^2}{c + \sigma^2_{f_{\backslash n}}}\right)^{-1} \\ \breve{\mu}_g &= \breve{\sigma}_g^2 \left(\frac{\mu_{g_{\backslash n}}}{\sigma^2_{g_{\backslash n}}} + \frac{y_n \mu_{f_{\backslash n}}}{c + \sigma^2_{f_{\backslash n}}}\right) \end{aligned} \tag{3.17}$$

and $\Phi(x)$ is the normal cumulative distribution function.

The first order moment $\hat{\boldsymbol{\mu}}_n = [\hat{\mu}_{f_n}, \hat{\mu}_{g_n}]^T$ can be separately found for each component. The expression for $\hat{\mu}_{g_n}$ can be written as

$$\hat{\mu}_{g_n} = \frac{1}{\hat{Z}_n} \iint g_n\; q_{\backslash n}(\phi_n)\; p(y_n|f_n, g_n)\; df_n\; dg_n = \frac{\breve{Z}}{\hat{Z}_n}\; \Phi\left(-\frac{\breve{\mu}_g}{\breve{\sigma}_g}\right) m_{2_t} \tag{3.18}$$

where m_{2_t} is the (non-centered) second order moment of the Gaussian truncated to the positive values of g_n with the mean and variance given in (3.17).

The expression for $\hat{\mu}_f$ is given by

$$\hat{\mu}_{f_n} = \frac{1}{\hat{Z}_n} \iint f_n\; q_{\backslash n}(\phi_n)\; p(y_n|f_n, g_n)\; df_n\; dg_n = \breve{\sigma}_f^2 \left(\frac{\mu_{f_{\backslash n}}}{\sigma^2_{f_{\backslash n}}} + \frac{y_n \hat{\mu}_{g_n}}{c}\right) \tag{3.19}$$

with $\breve{\sigma}_f^2 = (c^{-1} + \sigma_{f_{\setminus n}}^{-2})^{-1}$.

To calculate the covariance matrix $\hat{\Sigma}_n$, we only take into account the terms of the main diagonal $\hat{\sigma}_{f_n}^2$ and $\hat{\sigma}_{g_n}^2$, neglecting the cross-covariance terms out of the diagonal. To obtain $\hat{\sigma}_{f_n}^2$ and $\hat{\sigma}_{g_n}^2$, we first calculate the non-centered second order moments $\hat{m}_{2_{f_n}}$ and $\hat{m}_{2_{g_n}}$. For $\hat{m}_{2_{g_n}}$ we have

$$\hat{m}_{2_{g_n}} = \frac{1}{\hat{Z}_n} \iint g_n^2 \; q_{\setminus n}(\phi_n) \; p(y_n|f_n, g_n) \; df_n \; dg_n = \frac{\breve{Z}}{\hat{Z}_n} m_{3_t} \; \Phi\left(-\frac{\breve{\mu}_g}{\breve{\sigma}_g}\right) \tag{3.20}$$

where m_{3_t} is the non-centered third order moment of the Gaussian distribution with the parameters given in (3.17) truncated to the positive values of g_n.

The second order moment of f_n can be expressed as

$$\begin{aligned} \hat{m}_{2_{f_n}} &= \frac{1}{\hat{Z}_n} \iint f_n^2 \; q_{\setminus n}(\phi_n) \; p(y_n|f_n, g_n) \; df_n \; dg_n \\ &= \breve{\sigma}_f^2 + \frac{\hat{m}_{2_{g_n}} \breve{\sigma}_f^4 y_n^2}{c^2} + \frac{(\breve{\sigma}_f^2 \mu_{f_{\setminus n}})^2}{\sigma_{f_{\setminus n}}^4} + 2\frac{\breve{\sigma}_f^4 y_n \mu_{f_{\setminus n}} \hat{\mu}_{g_n}}{c + \sigma_{f_{\setminus n}}^2} \end{aligned} \tag{3.21}$$

Then, the expressions for $\hat{\sigma}_{f_n}^2$ and $\hat{\sigma}_{g_n}^2$ are given by

$$\begin{aligned} \hat{\sigma}_{f_n}^2 &= (\hat{m}_{2_{f_n}} - \hat{\mu}_{f_n}^2) \\ \hat{\sigma}_{g_n}^2 &= (\hat{m}_{2_{g_n}} - \hat{\mu}_{g_n}^2) \end{aligned} \tag{3.22}$$

Finally, with these calculated moments we re-estimate the parameters of the local likelihood approximation. For $\tilde{\sigma}_{f_n}^2$ and $\tilde{\sigma}_{g_n}^2$ we use the expression (2.35), whereas $\tilde{\mu}_{f_n}$ and $\tilde{\mu}_{g_n}$ can be calculated with (2.36). For $\tilde{Z}_n$ we have:

$$\begin{aligned} \tilde{Z}_n = \hat{Z}_n \sqrt{2\pi(\sigma_{f_{\setminus n}}^2 + \tilde{\sigma}_{f_n}^2)} \exp\left(\frac{(\mu_{f_{\setminus n}} - \tilde{\mu}_{f_n})^2}{2(\sigma_{f_{\setminus n}}^2 + \tilde{\sigma}_{f_n}^2)}\right) \times \\ \sqrt{2\pi(\sigma_{g_{\setminus n}}^2 + \tilde{\sigma}_{g_n}^2)} \exp\left(\frac{(\mu_{g_{\setminus n}} - \tilde{\mu}_{g_n})^2}{2(\sigma_{g_{\setminus n}}^2 + \tilde{\sigma}_{g_n}^2)}\right) \end{aligned} \tag{3.23}$$

Marginal likelihood

ML-II scheme is a common and computationally atractive procedure to find maximum likelihood estimates for hyperparameters by maximizing the logarithm of the marginal likelihood. In the case of EP, the evidence is approximated using the site functions instead of the real likelihood, as shown in (3.13). The expression for the log-evidence, needed to optimize hyperparameters, is given by

$$\begin{aligned}\log q(\mathbf{y}) = &-\frac{1}{2}\log|K_g+\tilde{\Sigma}_g| - \frac{1}{2}\log|K_f+\tilde{\Sigma}_f| \\ &-\frac{1}{2}(\boldsymbol{\mu}_0-\tilde{\boldsymbol{\mu}}_g)^T(K_g+\tilde{\Sigma}_g)^{-1}(\boldsymbol{\mu}_0-\tilde{\boldsymbol{\mu}}_g) \\ &-\frac{1}{2}\mathbf{y}^T(K_f+\tilde{\Sigma}_f)^{-1}\mathbf{y} + \sum_{n=1}^{N}\log(\hat{Z}_n) \\ &+\frac{1}{2}\sum_{n=1}^{N}\log(\sigma^2_{g_{\setminus n}}+\tilde{\sigma}^2_{g_n}) + \sum_{n=1}^{N}\frac{(\mu_{g_{\setminus n}}-\tilde{\mu}_{g_n})^2}{2(\sigma^2_{g_{\setminus n}}+\tilde{\sigma}^2_{g_n})} \\ &+\frac{1}{2}\sum_{n=1}^{N}\log(\sigma^2_{f_{\setminus n}}+\tilde{\sigma}^2_{f_n}) + \sum_{n=1}^{N}\frac{(\mu_{f_{\setminus n}}-\tilde{\mu}_{f_n})^2}{2(\sigma^2_{f_{\setminus n}}+\tilde{\sigma}^2_{f_n})}\end{aligned} \tag{3.24}$$

To calculate the derivatives of (3.24) with respect to the hyperparameters $\boldsymbol{\theta}$ we follow a treatment similar to [Rasmussen and Williams, 2006] for the case of EP for GP classification. The details of these calculations appear in Appendix C.2.

Predictive distribution

The predictive distribution for the output y_* given a new input $\mathbf{x}_*$ can be expressed as $p(y_*|\mathbf{x}_*)$ (omitting the conditioning on the training data). As it is not possible to calculate the exact expression, since we do not have the exact posterior, an approximation of this predictive distribution $q(y|\mathbf{x}_*)$ is calculated using the approximate posterior $q(\mathbf{f},\mathbf{g}|\mathbf{y})$ given by EP.

The first step involves the calculation of the predictive distributions for f_* and

g_*. The approximate predictive distribution $q(f_*)$ can be written as

$$q(f_*) = \int p(f_*|\mathbf{x}_*, \mathbf{f})\ q(\mathbf{f}, \mathbf{g}|\mathbf{y})\ d\mathbf{f}\ d\mathbf{g} = \mathcal{N}(f_*|\mu_{f_*}, \sigma^2_{f_*}) \tag{3.25}$$

where

$$\begin{aligned} \mu_{f_*} &= \mathbf{k}_{*f}^T (K_f + \tilde{\Sigma}_f)^{-1} \tilde{\boldsymbol{\mu}}_f \\ \sigma^2_{f_*} &= k_{f_{**}} - \mathbf{k}_{*f}{}^T \tilde{\Sigma}_f^{-1} \mathbf{k}_{*f} \end{aligned} \tag{3.26}$$

with $[\mathbf{k}_{*f}]_j = k_f(\mathbf{x}_*, \mathbf{x}_j)$ and $k_{f_{**}} = k_f(\mathbf{x}_*, \mathbf{x}_*)$.

In a similar way, the approximated predictive distribution $q(g_*)$ is given by

$$q(g_*) = \int p(g_*|\mathbf{x}_*, \mathbf{g})\ q(\mathbf{f}, \mathbf{g}|\mathbf{y})\ d\mathbf{f}\ d\mathbf{g} = \mathcal{N}(g_*|\mu_{g_*}, \sigma^2_{g_*}) \tag{3.27}$$

where

$$\begin{aligned} \mu_{g_*} &= \mathbf{k}_{*g}^T (K_g + \tilde{\Sigma}_g)^{-1} \tilde{\boldsymbol{\mu}}_g \\ \sigma^2_{g_*} &= k_{g_{**}} - \mathbf{k}_{*g}{}^T \tilde{\Sigma}_g^{-1} \mathbf{k}_{*g} \end{aligned} \tag{3.28}$$

Then, the approximate predictive distribution for y_* can be calculated analytically as

$$q(y_*) = \int p(y_*|f_*, g_*)\ q(f_*)\ q(g_*)\ df_*\ dg_* = Z_*(y_*)\ \breve{\mu}_{g_{*_t}}\ \Phi\left(-\frac{\breve{\mu}_{g_*}}{\breve{\sigma}_{g_*}}\right) \tag{3.29}$$

where

$$Z_*(y_*) = \mathcal{N}(\mu_{f_*}|\mu_{g_*} y_*, c + \sigma^2_{f_*} + \sigma^2_{g_*} y_*^2) \tag{3.30}$$

and $\breve{\mu}_{g_{*_t}}$ is the mean of the Gaussian $\mathcal{N}(g_*|\breve{\mu}_{g_*}, \breve{\sigma}^2_{g_*})$ truncated to the positive values of g_* with

$$\begin{aligned} \breve{\sigma}^2_{g_*} &= \left(\sigma^{-2}_{g_*} + \frac{y_*^2}{c + \sigma^2_{f_*}}\right)^{-1} \\ \breve{\mu}_{g_*} &= \breve{\sigma}^2_{g_*} \left(\frac{\mu_{g_*}}{\sigma^2_{g_*}} + \frac{y_* \mu_{f_*}}{c + \sigma^2_{f_*}}\right) \end{aligned} \tag{3.31}$$

The details of the calculation of this predictive distribution are included in Appendix C.3.

There is no analytical solution for the mean of $q(y_*)$. In general, the mean may not exist, as in the case of Cauchy distribution. To sidestep this problem we use the median as an estimator for the targets, which minimizes the mean absolute error. To do that, we use the expression of the cumulative distribution function for y_*

$$F_{y_*}(\alpha) = \int_{-\infty}^{\alpha} q(y_*)dy_* \tag{3.32}$$

calculating the value α that makes the cumulative distribution equals to 0.5. For the given $q(y_*)$, the solution follows a similar treatment than in the case of the cumulative distribution for the ratio of two correlated normal random variables, as described in [Hinkley, 1969], taking into account that we do not assume correlation between f and g and that we only take the positive part of g. Following this way, the cumulative distribution function is given by

$$F_{y_*}(\alpha) = \mathrm{L}\left(\frac{\mu_{g_*}\alpha - \mu_{f_*}}{a(\alpha)}, \frac{\mu_{g_*}}{\sigma_{g_*}}; \frac{\sigma_{g_*}\alpha}{a(\alpha)}\right) + \Phi\left(\frac{\mu_{g_*}}{\sigma_{g_*}}\right) \tag{3.33}$$

with

$$a(\alpha) = \sqrt{\sigma_{g_*}^2\alpha^2 + c + \sigma_{f_*}^2} \tag{3.34}$$

where $\mathrm{L}(h, k, \gamma)$ is the standard bivariate normal integral

$$\mathrm{L}(h, k, \gamma) = \frac{1}{2\pi\sqrt{1-\gamma^2}} \int_h^{\infty}\int_k^{\infty} \exp\left(-\frac{x^2 - 2\gamma xy + y^2}{2(1-\gamma^2)}\right) dx\ dy \tag{3.35}$$

If $\mu_{g_*}/\sigma_{g_*} \to \infty$, the cumulative distribution $F_{y_*}(\alpha)$ can be approximated by

$$F_{y_*}(\alpha) \to \Phi\left(\frac{\mu_{g_*}\alpha - \mu_{f_*}}{a(\alpha)}\right) \tag{3.36}$$

To obtain the predictive median m_{y_*}, we need the inverse cumulative distribution, i.e., $m_{y_*} = F_{y_*}^{-1}(1/2)$. Although calculation of $F_{y_*}^{-1}(\alpha)$ is not analytically tractable, we can approximate the solution using numerical root-finding methods as bisection or secant algorithms, for example.

Note that it is also possible to perform quantile estimation following the same treatment.

3.3 Experiments

3.3.1 A synthetic experiment

To assess the quality of the EP approximation compared to the golden standard MCMC, we have created a synthetic unidimensional data set with 150 samples according to the DGP model, selecting GP priors covariance functions for f and g to be squared exponentials (SE), defined as

$$k_{SE}(\mathbf{x}, \mathbf{x}') = \sigma_0^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell^2}\right) \tag{3.37}$$

with parameters $\ell_f = 0.7$ and $\sigma_{0_f}^2 = 9$ for the GP prior on f, and $\ell_g = 1.1$ and $\sigma_{0_g}^2 = 5$ for g. The prior mean of g is set to $\mu_0 = 3$, and $c = 4$.

To make inference with EP-DGP and MCMC-DGP, we set the hyperparameters to the known true values, avoiding the effect of hyperparameter learning, so that we can evaluate the quality of EP-DGP predictions with respect to the asymptotically unbiased MCMC estimates. To compare with a standard GP we cannot use the true DGP hyperparameters because the model is different. Instead, we search a proper set of hyperparameters using an ML-II implementation. For the MCMC-DGP implementation, we set a burn-in period of 5000 samples, i.e., the first 5000 drawn samples are not considered to estimate the posterior. After this burn-in period, the mean and variance of the posterior are approximated with 25000 samples drawn from the exact posterior.

To emphasize the high computational cost of MCMC-DGP with respect to the

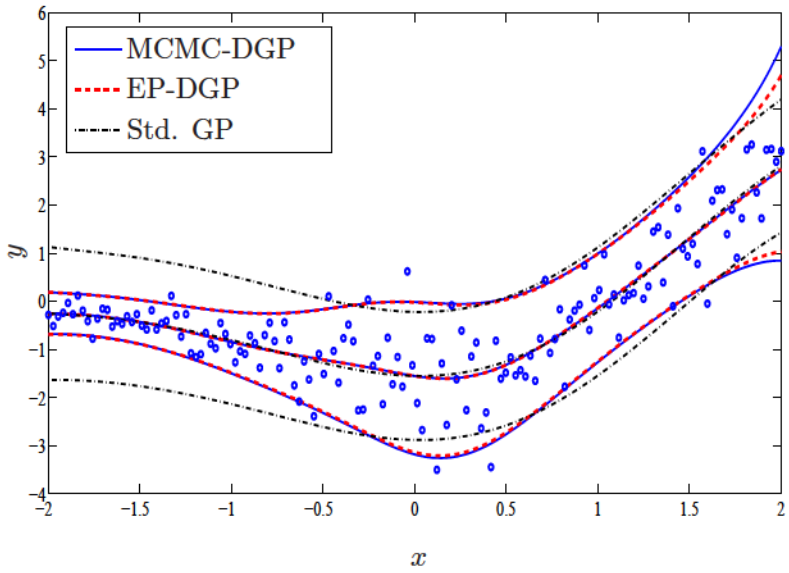


Figure 3.1: Posterior distributions over $y(x)$ for EP-DGP, MCMC-DGP, and the standard GP for a synthetic problem.

EP-DGP implementation[1], with these 150 samples, MCMC-DGP takes 90 seconds whereas EP-DGP takes only 2 seconds[2].

Figure 3.1 shows the posterior distributions over $y(x)$ for EP-DGP (dashed line), MCMC-DGP (continuous line), and standard GP (dash-dotted line). The median estimation is provided for EP-DGP and MCMC-DGP along with the 90% of mass probability bounded by quantiles 0.05 and 0.95. The estimated mean and twice the standard deviation are given for the standard GP prediction.

It can be appreciated that the EP-DGP posterior on $y(x)$ is very close to the exact solution provided by MCMC-DGP. Though not shown in Figure 3.1, the posteriors over latent functions f and g are very similar for the two compared methods,

[1]The Matlab implementation of EP-DGP and MCMC-DGP used in the experiments can be found at http://www.tsc.uc3m.es/%7Elmunoz/DGP.zip

[2]All the experiments have been conducted in a 4 GB computer with an Intel core i5 processor at 3.33 GHz using a Matlab implementation.

although differences can appear far from the registered data. Despite the effect of hyperparameters learning in the standard GP, it can be appreciated that the mean prediction is also similar to the other methods. However, the performance of the uncertainty prediction is clearly worse than DGP algorithms predictions, evidencing the limitations of the standard GP to deal with heteroscedastic noise.

3.3.2 Regression performance

In this subsection we present experimental results to evaluate the performance of DGP methods on several synthetic and real data sets. We compare the results with the standard GP, the state-of-the-art VHGPR [Lázaro-Gredilla and Titsias, 2011], the MAP Heteroscedastic GP (MAPHGP) described in [Quadrianto et al., 2009], the standard SVR [Drucker et al., 1997], and the Heteroscedastic NN model (H-NN) proposed in [Nix and Weigend, 1995].

As performance measures we use the Normalized MSE (NMSE)

$$\text{NMSE} = \frac{\sum_{j=1}^{n_*} (y_{*j} - \hat{y}_{*j})^2}{\sum_{j=1}^{n_*} (y_{*j} - \bar{y})^2} \tag{3.38}$$

the Normalized Mean Absolute Error (NMAE)

$$\text{NMAE} = \frac{\sum_{j=1}^{n_*} |y_{*j} - \hat{y}_{*j}|}{\sum_{j=1}^{n_*} |y_{*j} - \bar{y}|} \tag{3.39}$$

and the Negative Log-Predictive Density (NLPD)

$$\text{NLPD} = -\frac{1}{n_*} \sum_{j=1}^{n_*} \log p(y_{*j}|\mathcal{D}) \tag{3.40}$$

where n_* is the number of test samples, y_{*j} is the j-th test observation, $\hat{y}_{*j}$ is the predicted mean for that observation, and $\bar{y}$ is the mean of the training observations. Note that NLPD can take positive and negative values and that smaller values indicate better performance.

For VHGPR, we use an SE covariance function for the mean latent function and an SE covariance function plus noise for log-noise latent function. To initialize hyperparameters we previously train a standard GP following the procedure described in [Lázaro-Gredilla and Titsias, 2011].

The same covariance functions are applied in the case of MAPHGP, initializing the corresponding hyperparameters in the form detailed in [Quadrianto et al., 2009].

For DGP methods, we use an SE covariance function for $g(\mathbf{x})$ and a SE plus noise covariance function for $f(\mathbf{x})$. Assuming that $f(\mathbf{x})$ is a noisy function allows to establish a tradeoff between the heteroscedastic noise component modelled by $c/(g^+(\mathbf{x}))^2$ and the homoscedastic noise in $f(\mathbf{x})$.

To initialize the EP-DGP hyperparameters, we first train a standard GP with an SE plus noise covariance function. Then, the lengthscale hyperparameters of f and g are set to the lengthscale found by the standard GP, i.e., $\ell_f^{\text{EPDGP}} = \ell^{\text{GP}}$ and $\ell_g^{\text{EPDGP}} = \ell^{\text{GP}}$. For the amplitude and noise hyperparameters we first set $c = 4$ as a constant. As described previously, there is a degree of redundancy in the DGP model, so we do not need to learn c. With this consideration, we set $\mu_0 = 2/\sqrt{\sigma_n^{\text{GP}}}$, so that $c/\mu_0^2 = \sigma_n^{\text{GP}}$. For the power hyperparameter of g's covariance function we take $\sigma_{0_g}^{\text{EPDGP}} = \mu_0/\sqrt{10}$. Thus, we start from a quasi-homoscedastic initialization, so that nonstationarities have to be learned during the ML-II search. Finally, the power hyperparameter of f's covariance function is set to $\sigma_{0_f}^{\text{EPDGP}} = \mu_0\ \sigma_0^{\text{GP}}/c$, and the homoscedastic noise power hyperparameter to $\sigma_{n_f}^{\text{EPDGP}} = \sigma_{0_f}^{\text{EPDGP}}/2$.

We have not implemented any search procedure for MCMC-DGP hyperparameters. To evaluate the quality of EP-DGP approximation, we set MCMC-DGP hyperparameters equal to those obtained from the trained EP-DGP.

For SVR, we use a Radial Basis Function (RBF) kernel. The kernel width σ and the margin parameter C are set to the values that minimize the averaged test NMSE

over all the training/test splits created for each data set (as will described next). Notice that this induces a clear advantage for SVR designs. For a fair comparison, a cross validation for each split in each data set could be performed, for example. However, due to the high number of splits and the number of data sets used in the experiments, the computational burden of this methodology would be very high. The values of C are explored in the range $[2 \cdot 10^{-5}, 2 \cdot 10^{10}]$ with values of the form $2 \cdot 10^{p}$ with $-5 \leq p \leq 10$. For σ we explore also values of that form in the range $[2 \cdot 10^{-5}, 2 \cdot 10^{5}]$. To calculate the NLPD for SVR predictions, we assume a Gaussian distribution for the predictions with a noise power calculated as the MSE of the train predictions.

For the H-NN, we use two MLPs to estimate the mean of the targets and the log-noise, respectively, using the same number of neurons in the hidden layer (M) for both MLPs, as proposed in [Nix and Weigend, 1995]. We train the two MLPs using the Back-Propagation (BP) algorithm with the same methodology applied in [Nix and Weigend, 1995]. In the experiments, we have selected the value of M, explored in the range $[2, 40]$ for even values of M, that minimizes the averaged test NLPD over all the splits created for each data set (with 20 repetitions for each split). As in the case of the SVR, this induces an experimental advantage for H-NN designs with respect to the GP methods.

Unidimensional data sets

We will first consider five unidimensional data sets that have been much used in heteroscedastic regression literature, see e.g. [Kersting et al., 2007], [Lázaro-Gredilla and Titsias, 2011], [Muñoz-González et al., 2011]:

- *Nix* [Nix and Weigend, 1995] is a synthetic data set that consists on inputs uniformly spaced in the $[0, \pi]$ range and outputs given by $\sin(2.5x)\sin(1.5x)$ plus an heteroscedastic Gaussian noise with power varying as $0.01 + 0.25(1 -$

$\sin(2.5x))^2$.

- *Gol* [Goldberg et al., 1998] is a synthetic data set which considers inputs uniformly spaced in the $[0, 1]$ interval and outputs generated as $2\sin(2x)$ plus a Gaussian noise with linearly increasing standard deviation from 0.5 at $x = 0$ to 1.5 at $x = 1$.

- *Wah* [Yuan and Wahba, 2004] is a synthetic data set that consists on inputs uniformly spaced in the $[0, 1]$ range and outputs given by $2\ [\exp(-30(x - 0.25)^2) + \sin(\pi x^2)]$ plus Gaussian noise with power varying as $\exp(2\sin(2\pi x))$.

- *Caw* is the synthetic data set proposed in [Cawley et al., 2006], consisting on inputs uniformly spaced in the $[0, 1]$ interval, the outputs being the sign function of the inputs plus a Gaussian noise with standard deviation 0.1. Although the problem is homoscedastic, the steep change at $x = 0$ can be better modelled with a locally higher noise power.

- *Mot* is a real data set from [Silverman, 1985] with 133 samples which describe the acceleration force on a motorcycle helmet during an impact as a function of time.

For the synthetic problems, we have generated 300 independent runs with 100 samples drawn from the distributions described previously, i.e., different samples for each run. Then, a random split is performed, using 80% of the samples for training data and 20% of the samples for test. For data set *Mot* we have created 300 runs by randomly splitting its 133 samples into 80% for training and 20% for testing.

Table 3.1 shows the average performances in terms of NMSE, NMAE and NLPD for the standard GP, VHGPR, EP-DGP, MCMC-DGP, SVR, and H-NN. Statistical significant differences, measured with a Wilcoxon Rank-Sum test at the 5% significance level, are also shown in the table (a description of this nonparametric statistical test can be found in Appendix F).

Table 3.1: Experimental results on unidimensional data sets. Average NMSE, NMAE, and NLPD plus/minus one standard deviation are shown. Statistically significant differences are marked as • w.r.t. standard GP, ○ w.r.t. VHGPR, ◇ w.r.t. MCMC-DGP, ⋆ w.r.t. EP-DGP, ▷ w.r.t. SVR, and † w.r.t. H-NN. Significance is measured according to a Wilcoxon Rank-Sum test at the 5% level.

		Average NMSE	Average NMAE	Average NLPD
Nix:	Std. GP	0.650 ± 0.216 †	0.684 ± 0.155 †	0.979 ± 0.316
	VHGPR	0.573 ± 0.184 • †	0.608 ± 0.131 • †	0.468 ± 0.295 • ◇ ⋆ ▷ †
	MCMC-DGP	0.602 ± 0.220 • †	0.625 ± 0.143 • †	0.594 ± 0.267 • ▷ †
	EP-DGP	0.603 ± 0.221 • †	0.625 ± 0.143 • †	0.568 ± 0.285 • ▷ †
	SVR ($C = 20000; \sigma = 0.2$)	0.578 ± 0.181 • †	0.613 ± 0.127 • †	0.940 ± 0.355 •
	H-NN (M=22)	0.701 ± 0.169	0.715 ± 0.110	0.853 ± 0.313 • ▷
Gol:	Std. GP	0.727 ± 0.184 †	0.826 ± 0.118 †	1.519 ± 0.225
	VHGPR	0.724 ± 0.179 †	0.825 ± 0.118 †	1.457 ± 0.202 • ▷ †
	MCMC-DGP	0.727 ± 0.188 †	0.825 ± 0.121 †	1.444 ± 0.198 • ▷ †
	EP-DGP	0.727 ± 0.188 †	0.825 ± 0.121 †	1.445 ± 0.200 • ▷ †
	SVR ($C = 0.2; \sigma = 2$)	0.731 ± 0.155 †	0.830 ± 0.105 †	1.530 ± 0.241
	H-NN (M=26)	0.768 ± 0.188	0.850 ± 0.115	1.498 ± 0.207
Wah:	Std. GP	0.956 ± 0.179	0.951 ± 0.096	1.911 ± 0.311
	VHGPR	0.937 ± 0.146	0.941 ± 0.090	1.600 ± 0.287 • ▷ †
	MCMC-DGP	0.939 ± 0.205	0.938 ± 0.113	1.542 ± 0.274 • ○ ▷ †
	EP-DGP	0.940 ± 0.208	0.938 ± 0.114	1.542 ± 0.274 • ○ ▷ †
	SVR ($C = 200; \sigma = 0.2$)	0.908 ± 0.190 • ○ ⋆ †	0.915 ± 0.109 • ○ ◇ ⋆ †	1.893 ± 0.340
	H-NN (M=18)	0.971 ± 0.161	0.965 ± 0.094	1.672 ± 0.267 • ▷
Caw:	Std. GP	0.037 ± 0.030 ○ †	0.125 ± 0.034 ○ †	-0.135 ± 0.670 ▷
	VHGPR	0.063 ± 0.051 †	0.151 ± 0.053 †	-0.383 ± 0.509 • ▷ †
	MCMC-DGP	0.038 ± 0.033 ○ †	0.120 ± 0.034 • ○ ▷ †	-0.396 ± 0.131 • ○ ▷ †
	EP-DGP	0.037 ± 0.031 ○ †	0.120 ± 0.034 • ○ ▷ †	-0.584 ± 0.204 • ○ ▷ ◇ †
	SVR ($C = 20; \sigma = 20$)	0.037 ± 0.033 ○ †	0.126 ± 0.035 ○ †	0.183 ± 1.589
	H-NN (M=18)	0.078 ± 0.042	0.179 ± 0.045	-0.285 ± 0.235
Mot:	Std. GP	0.239 ± 0.081 †	0.454 ± 0.078 †	4.599 ± 0.154
	VHGPR	0.243 ± 0.080 †	0.454 ± 0.076 †	4.255 ± 0.195 • ◇ ⋆ ▷ †
	MCMC-DGP	0.249 ± 0.087 †	0.448 ± 0.080 †	4.359 ± 0.160 • ▷ †
	EP-DGP	0.248 ± 0.087 †	0.448 ± 0.079 †	4.323 ± 0.174 • ◇ ▷ †
	SVR ($C = 200; \sigma = 2$)	0.234 ± 0.092 ○ ◇ ⋆ †	0.441 ± 0.088 • ○ †	4.579 ± 0.191 •
	H-NN (M=10)	0.266 ± 0.082	0.483 ± 0.077	4.469 ± 0.164 • ▷

Analyzing the results of Table 3.1 in terms of NMSE and NMAE we can extract the following conclusions:

- The performance in terms of NMSE and NMAE is similar for all methods except the H-NN, which performs worse than the rest of the algorithms in all cases.
- We can appreciate a better performance of SVR in *Wah* and *Mot* data sets with respect to the GP methods. In *Nix*, the standard GP clearly performs worse than the other algorithms (except H-NN). Although in this data set SVR also performs better than the DGP methods, the improvement is not statistically significant.
- For data set *Caw*, VHGPR and H-NN perform sensibly worse than the rest of the methods in terms of NMSE. It can also be observed that EP-DGP and MCMC-DGP achieve significant better results than VHGPR and SVR in terms of NMAE.
- For data set *Mot*, it can be appreciated that there is some advantage for the SVR and the standard GPR for the NMSE measure. However, in terms of NMAE, DGP methods perform better than VHGPR and the standard GPR, though the SVR slightly outperforms EP-DGP and MCMC-DGP.

Comparing the performance in terms of NLPD in Table 3.1 (we remind that negative values are possible and smaller values are indicative of better performance), we can observe that:

- There is a very significant difference between the proposed DGP methods and the standard GPR and SVR in all data sets, even in *Caw*, where the problem is not really heteroscedastic, as described before. This shows the limitations of the standard GPR to estimate the predictive density when the noise is

nonstationary, and that the SVR cannot give good estimates of the noise power directly.

- The DGP methods outperform VHGPR in 3 of the 5 data sets, with statistical significant differences in *Wah*, for both MCMC-DGP and EP-DGP, and in *Caw* for EP-DGP. However, in problem *Gol*, the improvements of DGP methods are not statistically significant. In contrast, VHGPR performs better than both DGP algorithms for data sets *Nix* and *Mot*.

- Although the H-NN method outperforms GPR and SVR in terms of NLPD, the performance with respect to the other heteroscedastic GP methods is clearly worse.

Additionally, it is also important to note that the results of EP-DGP and MCMC-DGP are very similar for most the data sets. Although there are some differences in terms of NLPD for *Nix*, *Caw* and *Mot*, these are not statistically significant, and it can be observed that the results in terms of NMSE and NMAE are very similar. These results validate the quality of the EP posterior approximation compared to the exact posterior obtained with the MCMC implementation, using the same set of hyperparameters than EP-DGP.

With the code implementation of MAPHGP kindly provided by the authors of [Quadrianto et al., 2009] we were not able to obtain good experimental results for the unidimensional data sets used before, since the MAPHGP did not converge or performed really poor for some of the splits in the 5 unidimensional data sets. In these conditions, we have performed a similar experiment but selecting 300 splits for each data set where MAPHGP converged and had a competitive performance. Then, we compared the results with EP-DGP on those selected splits, observing an experimental advantage to MAPHGP with respect to EP-DGP. The results of this experiment are shown in Table 3.2, providing the average NMSE, NMAE, and NLPD along with the standard deviation for EP-DGP and MAPHGP for the 5

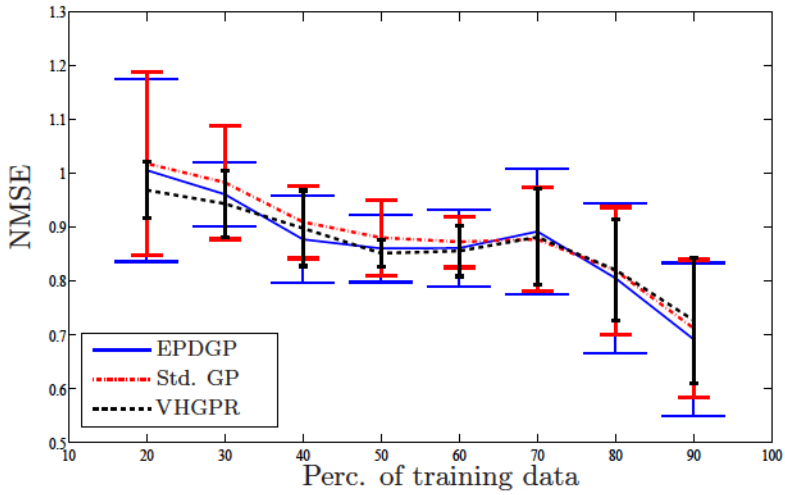
unidimensional data sets. Statistically significant differences are measured using the Wilcoxon Rank-Sum test at the 5% level.

Table 3.2: Experimental test results on unidimensional data sets for MAPHGP compared to EP-DGP. Average NMSE, NMAE, and NLPD plus/minus one standard deviation are shown. Statistically significant differences are marked as • w.r.t. MAPHGP and ⋆ w.r.t. EP-DGP. Significance is measured according to a Wilcoxon Rank-Sum test at the 5% level.

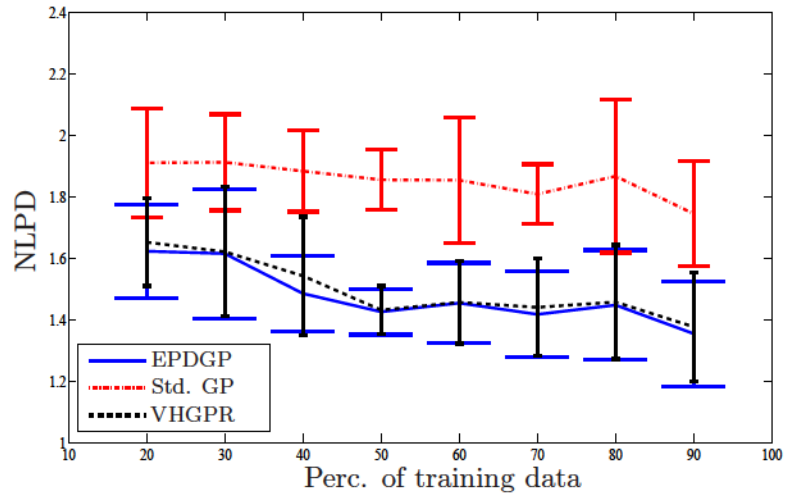
		MAPHGP	EP-DGP
Nix:	NMSE	0.578 ± 0.190	0.596 ± 0.208
	NMAE	0.621 ± 0.135	0.621 ± 0.127
	NLPD	0.597 ± 0.439	0.571 ± 0.221
Gol:	NMSE	0.739 ± 0.187	0.737 ± 0.187
	NMAE	0.832 ± 0.121	0.831 ± 0.122
	NLPD	1.531 ± 0.252	1.430 ± 0.188 •
Wah:	NMSE	0.953 ± 0.124	0.958 ± 0.213
	NMAE	0.957 ± 0.082	0.951 ± 0.112
	NLPD	1.718 ± 0.293	1.530 ± 0.248 •
Caw:	NMSE	0.099 ± 0.101	0.041 ± 0.038 •
	NMAE	0.185 ± 0.095	0.122 ± 0.039 •
	NLPD	-0.266 ± 0.436	-0.386 ± 0.144 •
Mot:	NMSE	0.246 ± 0.095	0.248 ± 0.093
	NMAE	0.463 ± 0.082	0.452 ± 0.080
	NLPD	4.314 ± 0.264 ⋆	4.360 ± 0.153

We can appreciate that the performance in terms of NMSE and NMAE is similar for both methods, although EP-DGP outperforms MAPHGP in *Caw* with statistical significance. However, in terms of NLPD, EP-DGP performs better than MAPHGP in 4 of the 5 data sets (in 3 of them with statistically significant difference), whereas

MAPHGP only outperforms EP-DGP in *Mot*.



(a)



(b)

Figure 3.2: Experiment on Wahba problem with 200 samples varying the percentage of training data. Average NMSE (a) and NLPD (b) as a function of the percentage of training data for EP-DGP (continous line and wide error bars), standard GP (dashed-dotted line and medium-size error bars), and VHGPR (dashed line and narrow error bars), along with the 2σ error bars.

To analyze the degradation of the performance and the effect of the hyperparameters learning of EP-DGP when reducing the number of training samples, we have conducted an experiment with data set *Wah*. With 200 samples, we have generated random splits varying the number of training samples from 20% to 90% with steps of 10%, with 10 independent runs at each step. Figure 3.2 shows the results of this experiment, comparing the performance of the standard GP, VHGPR, and EP-DGP in terms of NMSE and NLPD.

It can be appreciated than the degradation of the performance in terms of NMSE is quite similar for the three methods, although EP-DGP and VHGPR have more hyperparameters to learn than the standard GP, because they have two latent functions with their corresponding covariance functions for the two GP priors. In terms of NLPD, there is a noticeable difference between the standard GP and both heteroscedastic methods. However, it seems that standard GP degradation is smoother than the rest of the algorithms. It can also be observed that EP-DGP slightly outperforms VHGPR in all cases, corroborating the results shown in Table 3.1 for 80 training samples.

Multidimensional data sets

The second part of the experiments shows the performance of DGP methods on multivariate data sets. We have selected the following 7 real problems whose stationary or nonstationary nature is a priori unknown:

- The prostate cancer (*Can*) problem is a data set to estimate the level of prostate specific antigen (PSA) with 8 clinical measures in 97 men who were about to receive a radical prostatectomy [Stamey et al., 1989].
- *Ozo* is a three dimensional data set to estimate ozone concentrations with measures of temperature, radiation, and wind speed [Hastie et al., 2009].

- Diabetes data set (*Dia*) consists on 10 baseline variables from 442 diabetes patients to estimate a quantitative measure of disease progression one year after [Efron et al., 2004].
- The Boston housing (*Hou*) is a frequently used regression problem in a 13-dimensional feature space to estimate the mean value of housing in Boston metropolitan area [Harrison and Rubinfeld, 1978].
- The concrete compressive strength data set (*Con*) consits on 10 features used to model the compressive strength of high performance concrete [Yeh, 1998].
- Abalone (*Aba*) consists on 8 phisical measurements to predict the age of abalones [Bache and Lichman, 2014].
- Parkinson Telemonitoring data set (*Par*) contains 21 biomedical variables to estimate the motor Unified Parkinson's Disease Rating Scale of people with early-stage Parkinson's disease [Tsanas et al., 2010].

Table 3.3 shows the number of features of this 7 multivariate data sets along with the number of training and test samples.

Table 3.3: Characteristics of multidimensional data sets used for the experiments.

Data set	Dimension	# Training samples	# Test samples
Can	8	78	19
Ozo	3	89	22
Dia	10	221	221
Hou	13	253	253
Con	8	515	515
Aba	8	1044	3133
Par	21	1000	4875

We have made 300 random splits with 80% training samples and 20% test samples for *Can* and *Ozo*. For data sets *Dia*, *Hou*, and *Con*, we have also made 300 random splits with 50% training and test samples. We have selected a bigger percentage of training samples for *Can* and *Ozo* because those data sets are sensibly smaller and hyperparameter learning problems may arise for all the GP based methods used in the experiments. For the biggest data sets, *Aba* and *Par*, we have only made one split.

The results of the experiments for the data sets *Can*, *Ozo*, *Dia*, *Hou*, and *Con*, are presented in Table 3.4. For MCMC-DGP, we only show results for *Can* and *Ozo* as the computational burden for the rest of the data sets is very high. We have also performed a Wilcoxon Rank-Sum test to assess statistical significance at the 5% level.

With the code implementation of MAPHGP provided by the authors of [Quadrianto et al., 2009] we have not been able to produce satisfactory results for these multidimensional data sets.

It can be appreciated that H-NN clearly subperforms the rest of the methods used in the experiments in all data sets. This shows that the H-NN model proposed in [Nix and Weigend, 1995] is not suitable for multidimensional data sets.

In terms of NMSE, it can be observed that:

- The SVR is worse than the rest of GP based methods with statistical significance in 4 of the 5 data sets.
- DGP methods outperform VHGPR in 3 data sets and the standard GP in 2 with statistical significance.
- In none of the data sets DGP algorithms perform worse than the other GP based methods.

Table 3.4: Experimental test results on multidimensional data sets. Average NMSE, NMAE, and NLPD plus/minus one standard deviation are shown. Statistically significant differences are marked as • w.r.t. standard GP, ○ w.r.t. VHGPR, ◇ w.r.t. MCMC-DGP, ⋆ w.r.t. EP-DGP, ▷ w.r.t. SVR, and † w.r.t. H-NN. Significance is measured according to a Wilcoxon Rank-Sum test at the 5% level.

		Average NMSE	Average NMAE	Average NLPD
Can:	Std. GP	0.440 ± 0.191 †	0.634 ± 0.129 †	1.146 ± 0.183 †
	VHGPR	0.440 ± 0.191 †	0.634 ± 0.129 †	1.146 ± 0.183 †
	MCMC-DGP	0.438 ± 0.181 †	0.634 ± 0.122 †	1.117 ± 0.185 • ○†
	EP-DGP	0.438 ± 0.180 †	0.633 ± 0.121 †	1.117 ± 0.187 • ○†
	SVR ($C = 2; \sigma = 0.02$)	0.405 ± 0.148 • ○ ◇ ⋆†	0.618 ± 0.111 †	1.138 ± 0.233 †
	H-NN (M=2)	1.267 ± 0.887	0.938 ± 0.109	1.509 ± 0.160
Ozo:	Std. GP	0.281 ± 0.091 ▷ †	0.484 ± 0.085 †	4.323 ± 0.271 ▷ †
	VHGPR	0.282 ± 0.083 ▷ †	0.477 ± 0.079 †	4.156 ± 0.209 • ▷†
	MCMC-DGP	0.258 ± 0.079 • ○ ▷ †	0.462 ± 0.079 • ○ ▷ †	4.074 ± 0.143 • ○ ▷ †
	EP-DGP	0.258 ± 0.079 • ○ ▷ †	0.462 ± 0.079 • ○ ▷ †	4.071 ± 0.151 • ○ ▷ †
	SVR ($C = 200; \sigma = 0.2$)	0.300 ± 0.094 †	0.486 ± 0.085 †	4.443 ± 0.530 †
	H-NN (M=2)	0.571 ± 0.094	0.686 ± 0.068	4.552 ± 0.184
Dia:	Std. GP	0.505 ± 0.034 ▷ †	0.672 ± 0.025 ▷ †	5.429 ± 0.034 ▷ †
	VHGPR	0.505 ± 0.034 ▷ †	0.672 ± 0.025 ▷ †	5.424 ± 0.034 ▷ †
	EP-DGP	0.505 ± 0.034 ▷ †	0.671 ± 0.024 ▷ †	5.413 ± 0.035 • ○ ▷ †
	SVR ($C = 200; \sigma = 0.002$)	0.518 ± 0.029 †	0.687 ± 0.023 †	5.442 ± 0.029 †
	H-NN (M=8)	0.554 ± 0.032	0.706 ± 0.022	5.470 ± 0.036
Hou:	Std. GP	0.152 ± 0.035 ○ ▷†	0.352 ± 0.022 ▷ †	2.624 ± 0.122 ▷ †
	VHGPR	0.164 ± 0.034 ▷ †	0.352 ± 0.022 ▷ †	2.575 ± 0.146 • ▷†
	EP-DGP	0.152 ± 0.036 ○ ▷†	0.336 ± 0.022 • ○ ▷ †	2.408 ± 0.069 • ○ ▷ †
	SVR ($C = 200; \sigma = 0.02$)	0.177 ± 0.042 †	0.356 ± 0.024 †	3.126 ± 0.500 †
	H-NN (M=4)	0.832 ± 0.354	0.809 ± 0.039	3.458 ± 0.311
Con:	Std. GP	0.132 ± 0.014 ○ ▷†	0.326 ± 0.014 ▷ †	3.162 ± 0.041 ▷ †
	VHGPR	0.134 ± 0.013 ▷ †	0.327 ± 0.014 ▷ †	3.088 ± 0.042 • ▷†
	EP-DGP	0.123 ± 0.015 • ○ ▷ †	0.310 ± 0.014 • ○ ▷ †	3.049 ± 0.041 • ○ ▷ †
	SVR ($C = 2000; \sigma = 0.02$)	0.158 ± 0.017 †	0.353 ± 0.016 †	3.391 ± 0.106 †
	H-NN (M=8)	0.364 ± 0.030	0.568 ± 0.017	3.714 ± 0.054

In a similar way, in terms of NMAE it can be appreciated that:

- DGP algorithms perform significantly better than the standard GP and VHGPR in *Ozo*, *Hou*, and *Con*. For *Can* and *Dia*, the results are almost equivalent for the three methods.
- Again, SVR is worse than the rest of the compared methods in 4 of the 5 data sets.

In terms of NLPD, we notice that:

- DGP methods perform better than VHGPR, SVR, and the standard GP with statistical significance in all data sets except for SVR in *Can*, where, although EP-DGP and MCMC-DGP have better results, there are not statistically significant differences.
- VHGPR only improves the standard GP results in *Ozo*, *Hou* and *Con*.
- GP based methods outperform SVR in 4 of the 5 data sets with statistical significance. In *Can*, though SVR performs slightly better than the standard GP and VHGPR, DGP methods have better results than SVR.

It can be also appreciated that EP-DGP and MCMC-DGP results for *Can* and *Ozo* are very similar with respect to NMSE, NMAE, and NLPD measures. As in the case of the experiments with unidimensional data sets, these results support the quality of the EP approximation compared to the exact posterior given by MCMC-DGP.

From the results presented in Table 3.4, we can conclude that although the heteroscedastic nature of the selected problems is a priori unknown, VHGPR and DGP methods never perform worse than the standard GP, even for problems that seem

to be more homoscedastic, as data sets *Can* and *Dia*. These results support the advantages of the evaluated heteroscedastic methods and show the limitations of the standard GPR when the problem is not stationary.

Table 3.5: Experimental test results on multidimensional data sets *Aba* and *Par*. NMSE, NMAE, and NLPD are provided for the standard GP, VHGPR, EP-DGP, and SVR.

		NMSE	NMAE	NLPD
Aba:	Std. GP	0.428	0.634	2.167
	VHGPR	0.429	0.634	2.068
	EP-DGP	0.459	0.639	2.039
	SVR ($C = 2 \cdot 10^6; \sigma = 2 \cdot 10^{-5}$)	0.446	0.640	2.202
Par:	Std. GP	0.184	0.384	−1.968
	VHGPR	0.178	0.381	−2.014
	EP-DGP	0.192	0.389	−2.023
	SVR ($C = 200; \sigma = 2 \cdot 10^{-4}$)	0.227	0.492	−1.719

The experimental results for the data sets *Aba* and *Par* are shown in Table 3.5. In these data sets, we have used a squared exponential covariance function with Automatic Relevance Determination (ARD), defined as

$$k_{ARD}(\mathbf{x}, \mathbf{x}') = \sigma_0^2 \exp\left(-\sum_{i=1}^{D} \frac{\left(\mathbf{x}^{(i)} - \mathbf{x}'^{(i)}\right)^2}{2\ell_i^2}\right) \tag{3.41}$$

where $\mathbf{x}^{(i)}$ is the i-th component of the input sample $\mathbf{x}$. This covariance function is more suitable for data sets with a significant number of training samples compared to the number of features. It can be observed that EP-DGP outperforms the rest of the compared methods in terms of NLPD. Although VHGPR and the standard GP achieve a better NMSE than EP-DGP, the NMAE is quite similar for the three methods. The SVR does not present competitive advantages in any case.

3.3.3 Time scalability experiment

We have performed an additional experiment with data set *Wah* to measure the training time for the standard GP, VHGPR, EP-DGP, and MCMC-DGP for different sizes of the training set.

First, we learn the set of hyperparameters for all the methods using 300 training samples. Then, we measure the training time (avoiding the hyperparameters learning) varying the number of training samples from 50 to 300 with steps of 50 samples. For a given number of training samples, we generate 20 random training sets. The average training time with the corresponding error bars for the different sizes of the training set are shown in Fig. 3.3 for the standard GP (continuous line and wider error bars), VHGPR (dashed line), EP-DGP (dashed-dotted line), and MCMC-DGP (dotted line).

It can be observed a very high computational cost of MCMC-DGP, compared with the other methods. Although EP-DGP is slower than VHGPR and the standard GP, its computational burden is affordable. It can also be noticed that the slope of the logarithm of the training time as a function of the logarithm of the training set size is similar. This supports that the complexity of all these methods scales in the same form, concretely $\mathcal{O}(N^3)$.

All the GP models discussed in this chapter fall within the same complexity category: They all require $\mathcal{O}(N^3)$ computation time (which arises from the inversion of $N \times N$ matrices, or, alternatively, from the numerically more stable Cholesky decomposition of such matrices) and $\mathcal{O}(N^2)$ space (which is required to store such matrices). However, their speeds differ by the constant factor that vanishes when using the $\mathcal{O}(\cdot)$ notation. The difference in this constant factor is what makes MCMC unsuitable for application on medium-sized data sets, while making the alternative approximations suitable for data sets with a few thousands of data points. The

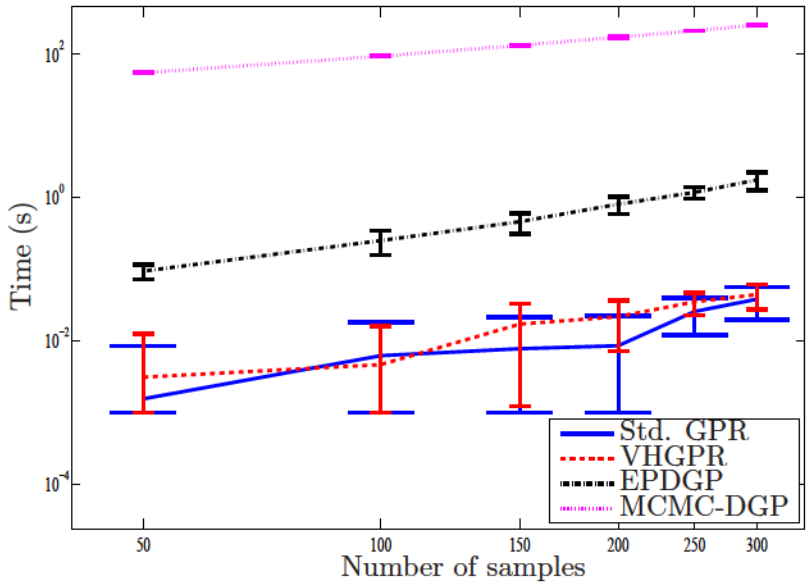


Figure 3.3: Results of the training time experiment using *Wah* data set with different sizes for the training set.

empirical results shown in Figure 3.3 provide a quantitative measurement of this speed-up factor for the different GP based algorithms.

3.4 Conclusions

In this chapter we have introduced a divisive GP (DGP) model to carry out nonstationary regression including heteroscedastic noise cases. The likelihood log-concavity of the proposed model leads to a unimodal posterior that favors EP Gaussian approximation, in contrast to the heteroscedastic model used in [Kersting et al., 2007, Quadrianto et al., 2009, Lázaro-Gredilla and Titsias, 2011, Muñoz-González et al., 2011] or the GP product model proposed in [Adams and Stegle, 2008].

The MCMC with ESS implementation provides an accurate exact posterior for the DGP, although the computational burden is very high. Yet the EP posterior approximation reduces the computational cost offering a similar performance for regression tasks.

The experimental results showed that the homoscedascity assumption of the standard GP or the SVR can be too restrictive for many real problems. Therefore, DGP methods improve standard GP and SVR predictions (mainly in terms of NLPD) when the problem is heteroscedastic, whereas the performance is similar if the data is stationary. Moreover, EP-DGP results are very competitive when comparing with other heteroscedastic methods such as VHGPR [Lázaro-Gredilla and Titsias, 2011] and MAPHGP [Quadrianto et al., 2009], outperforming both algorithms in several problems.

The main drawback of the proposed approach is the higher computational burden of the proposed EP approximation compared to the standard GP or VHGPR, despite all these methods scale in time as $\mathcal{O}(N^3)$. However, as shown in the experiments, EP-DGP can be used in practice for data sets with a few thousands samples.

Chapter 4

Divisive Gaussian processes for nonstationary regression with the Laplace approximation

Although the EP approximation to make inference on the DGP model presented in Chapter 3 performs reasonably well on nonstationary and heteroscedastic problems, the computational complexity of EP limits the use of the proposed method for big data sets. On the other hand, although EP is conjectured to converge for log-concave likelihoods, as it is the case of the DGP model, there is no formal proof of convergence. To solve this limitations, in this chapter we propose to apply the Laplace approximation for DGP (L-DGP). The log-concave likelihood of DGP model combined with the GP priors provides a log-concave (and unimodal) posterior, which ensures the convergence of L-DGP to a unique maximum.

The simplicity of the Laplace approximation compared to EP leads in some cases to poor posterior approximations as, for example, in GP for classification. However,

the characteristics of the DGP model, where the posterior is expected to have quite a Gaussian shape, makes the Laplace method a suitable alternative to provide accurate posterior approximations, similar to those provided by EP-DGP. Moreover, the Laplace method reduces the computational burden with respect to similar approaches like EP or variational Bayes. However, the main drawback is the $\mathcal{O}(N^3)$ time scalability, as it is the case for many GP based methods.

The rest of the chapter is organized as follows. In Section 4.1 we describe L-DGP inference procedure. Then, in Section 4.2 we show the experimental results that validate the utility of the proposed Laplace approximation compared to EP-DGP and other competing methods such as the standard GP and VHGPR. Finally, a summary and the main conclusions are presented in Section 4.3.

4.1 Inference with the Laplace approximation

As explained in Chapter 2, the Laplace method aims to approximate the posterior $p(\mathbf{f}, \mathbf{g}|\mathbf{y})$ with a Gaussian distribution $q(\mathbf{f}, \mathbf{g}|\mathbf{y})$ doing a second order Taylor expansion around the maximum of the log-posterior

$$q(\boldsymbol{\phi}|\mathbf{y}) = \mathcal{N}(\boldsymbol{\phi}|\hat{\boldsymbol{\phi}}, A^{-1}) \propto \exp\left(-\frac{1}{2}(\boldsymbol{\phi} - \hat{\boldsymbol{\phi}})^T A(\boldsymbol{\phi} - \hat{\boldsymbol{\phi}})\right) \tag{4.1}$$

where $\boldsymbol{\phi} = [f, g]$, $\hat{\boldsymbol{\phi}} = \arg\max_{\boldsymbol{\phi}} p(\boldsymbol{\phi}, \mathbf{y})$, and $A = -\nabla\nabla \log p(\boldsymbol{\phi}, \mathbf{y})|_{\boldsymbol{\phi}=\hat{\boldsymbol{\phi}}}$ is the Hessian of the log-posterior at $\boldsymbol{\phi} = \hat{\boldsymbol{\phi}} = [\hat{\mathbf{f}}, \hat{\mathbf{g}}]$.

To simplify the calculations with the Laplace approximation, we have considered an equivalent model to the divisive model described previously, placing μ_0, the offset of the GP prior of latent function g, in the likelihood, i.e.,

$$p(y_n|f_n, g_n) = \mathcal{N}(y_n|f_n/g_n'^{+}, c/(g_n'^{+})^2) \tag{4.2}$$

with $g_n' = g_n + \mu_0$. Therefore, the GP prior of the latent function g for this equivalent model has zero mean, i.e., $g(\mathbf{x}) \sim \mathcal{GP}(0, k_g(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}_g))$.

To maximize the log-posterior with respect to the latent functions at the training points only the GP priors and the likelihood are needed, since the evidence $p(\mathbf{y})$ is independent on $\mathbf{f}$ and $\mathbf{g}$. Hence, the expression to be maximized reduces to

$$\Psi(\phi) = \log p(\mathbf{y}|\phi) + \log p(\phi) = \log p(\mathbf{y}|\phi) - \frac{1}{2}\phi^T K^{-1}\phi - N\log 2\pi \tag{4.3}$$

where K is the $2N$-square block diagonal matrix

$$K = \begin{bmatrix} K_f & 0_N \\ 0_N & K_g \end{bmatrix} \tag{4.4}$$

where K_f and K_g are the covariance matrices of the GP priors on f and g, respectively, and 0_N is a N-square matrix with all its elements equal to zero.

Making $\nabla\Psi(\phi) = 0$, we obtain the following self-consistent equation at the maximum of $\nabla\Psi(\phi)$

$$\hat{\phi} = K(\nabla \log p(\mathbf{y}|\hat{\phi})) \tag{4.5}$$

To search the maximum, the Newton method can be applied as in (2.22). Then, the update equation can be expressed as

$$\phi_{\text{new}} = (K^{-1} + W)^{-1}(W\phi + \nabla \log p(\mathbf{y}|\phi)) \tag{4.6}$$

where $W = -\nabla\nabla \log p(\mathbf{y}|\phi)$ is the negative Hessian of the likelihood, which yields the block diagonal matrix

$$W = \begin{bmatrix} W_f & W_{fg} \\ W_{fg} & W_g \end{bmatrix} \tag{4.7}$$

The n-th diagonal elements of each of the three diagonal matrices W_f, W_g, and W_{fg} are given by

$$[W_f]_{nn} = -\frac{\partial^2 \log p(y_n|f_n, g_n)}{\partial f_n^2} = \frac{1}{c} \tag{4.8}$$

$$[W_g]_{nn} = -\frac{\partial^2 \log p(y_n|f_n, g_n)}{\partial g_n^2} = \begin{cases} \dfrac{1}{g_n'^2} + \dfrac{y_n^2}{c}, & g_n' > 0 \\ 0, & g_n' \leq 0 \end{cases} \tag{4.9}$$

$$[W_{fg}]_{nn} = -\frac{\partial^2 \log p(y_n|f_n, g_n)}{\partial f_n \partial g_n} = \begin{cases} -\frac{y_n}{c}, & g'_n > 0 \\ 0, & g'_n \leq 0 \end{cases} \tag{4.10}$$

The gradient of the likelihood, $\nabla \log p(\mathbf{y}|\phi)$, is the vector of length $2N$

$$[\nabla \log p(\mathbf{y}|\phi)]_n = \frac{\partial \log p(y_n|f_n, g_n)}{\partial f_n} = \frac{g'^{+}_n y_n - f_n}{c} \tag{4.11}$$

for $1 \leq n \leq N$, and

$$[\nabla \log p(\mathbf{y}|\phi)]_n = \frac{\partial \log p(y_{n'}|f_{n'}, g_{n'})}{\partial g_{n'}} = \begin{cases} \frac{1}{g'_{n'}} - \frac{y^2_{n'} g'_{n'} - y_{n'} f_{n'}}{c}, & g'_{n'} > 0 \\ 0, & g'_{n'} \leq 0 \end{cases} \tag{4.12}$$

for $N < n \leq 2N$, with $n' = n - N$.

As the likelihood is (jointly) log-concave on f and g, the Hessian results in a negative definite matrix. Thus, $\Psi(\phi)$ is concave and the Laplace method converges to a unique maximum.

4.1.1 Approximate Marginal Likelihood

As explained in Chapter 2, ML-II is the most common (and computationally atractive) procedure to find the set of hyperparameters, consisting on maximizing the logarithm of the marginal likelihood with respect to those hyperparameters. As in the case of the EP approximation described in Chapter 3, since it is not possible to obtain an analytical expression for the exact log-evidence under the DGP model, we have to provide an approximate expression using the posterior approximation given by the Laplace method. The approximate log-evidence can be written as

$$\log q(\mathbf{y}) = -\frac{1}{2}\hat{\phi}^T K^{-1} \hat{\phi} + \log p(\mathbf{y}|\hat{\phi}) - \frac{1}{2}\log |B| \tag{4.13}$$

with

$$|B| = |K| \cdot |K^{-1} + W| \tag{4.14}$$

The determinant of B can be computed more efficiently applying the Cholesky factorization to W, so that $W = LL^T$ (see Appendix B.4), and the matrix determinant lemma (see Appendix B.2) resulting in

$$|B| = |I + L^T K L| \tag{4.15}$$

The derivatives of the approximate log-evidence in (4.13) with respect to the hyperparameters are shown in Appendix D.

4.1.2 Predictive distribution

Given a test sample $\mathbf{x}_*$, the predictive distribution for the corresponding output y_* can be expressed as $p(y_*|\mathbf{x}_*)$ (omitting the conditioning on the training data). Since we do not have an exact expression for the posterior on the training data, we have to calculate an approximate predictive distribution $q(y_*|\mathbf{x}_*)$ using the posterior approximation $q(\mathbf{f}, \mathbf{g}|\mathbf{y})$.

To calculate $q(y_*|\mathbf{x}_*)$ we first need to calculate the predictive distributions for latent functions f and g for a test sample $\mathbf{x}_*$, i.e., f_* and g_*. Thus, the approximate predictive distribution $q(f_*)$ can be written as

$$q(f_*) = \int p(f_*|\mathbf{x}_*, \mathbf{f}) q(\mathbf{f}, \mathbf{g}|\mathbf{y}) d\mathbf{f} d\mathbf{g} = \mathcal{N}(f_*|\mu_{f_*}, \sigma^2_{f_*}) \tag{4.16}$$

where

$$\begin{aligned} \mu_{f_*} &= \mathbf{k}_{*f}^T (\nabla \log p(\mathbf{y}|\phi)|_f) \\ \sigma^2_{f_*} &= k_{f_{**}} - \mathbf{k}_{*f}^T (K_f + W_f^{-1})^{-1} \mathbf{k}_{*f} \end{aligned} \tag{4.17}$$

with $[\mathbf{k}_{*f}]_j = k_f(\mathbf{x}_*, \mathbf{x}_j)$, $k_{f_{**}} = k_f(\mathbf{x}_*, \mathbf{x}_*)$, and $\nabla \log p(\mathbf{y}|\phi)|_f$ are the first N elements of vector $\nabla \log p(\mathbf{y}|\phi)$ (those corresponding to the partial derivatives with respect to f).

In a similar way, the approximate predictive distribution $q(g_*)$ is given by

$$q(g_*) = \int p(g_*|\mathbf{x}_*, \mathbf{g}) q(\mathbf{f}, \mathbf{g}|\mathbf{y}) d\mathbf{f} d\mathbf{g} = \mathcal{N}(g_*|\mu_{g_*}, \sigma^2_{g_*}) \tag{4.18}$$

where

$$\begin{aligned} \mu_{g_*} &= \mathbf{k}_{*g}^T (\nabla \log p(\mathbf{y}|\boldsymbol{\phi})|_g) \\ \sigma^2_{g_*} &= k_{g_{**}} - \mathbf{k}_{*g}^T (K_g + W_g^{-1})^{-1} \mathbf{k}_{*g} \end{aligned} \tag{4.19}$$

and $\nabla \log p(\mathbf{y}|\boldsymbol{\phi})|_g$ is a vector containing the last N elements of $\nabla \log p(\mathbf{y}|\boldsymbol{\phi})$.

We have neglected the covariance term $\sigma_{f_* g_*}$ to simplify the predictive distribution for y_*, as the correlation between latent functions is expected to be an artifact. Then, the approximate predictive distribution for y_* can be calculated as

$$q(y_*) = \int p(y_*|f_*, g_*) q(f_*) q(g_*) df_* dg_* = Z_*(y_*)\ \tilde{\mu}_{g_{*_t}}\ \Phi\left(-\frac{\tilde{\mu}_{g_*}}{\tilde{\sigma}_{g_*}}\right) \tag{4.20}$$

where

$$Z_*(y_*) = \mathcal{N}(\mu_{f_*}|\mu_{g_*} y_*, c + \sigma^2_{f_*} + \sigma^2_{g_*} y_*^2) \tag{4.21}$$

and $\tilde{\mu}_{g_{*_t}}$ is the mean of the Gaussian $\mathcal{N}(g_*|\tilde{\mu}_{g_*}, \tilde{\sigma}^2_{g_*})$ truncated to the positive values of g_* with

$$\begin{aligned} \tilde{\sigma}^2_{g_*} &= \left(\sigma^{-2}_{g_*} + \frac{y_*^2}{c + \sigma^2_{f_*}}\right)^{-1} \\ \tilde{\mu}_{g_*} &= \tilde{\sigma}^2_{g_*} \left(\frac{\mu_{g_*}}{\sigma^2_{g_*}} + \frac{y_* \mu_{f_*}}{c + \sigma^2_{f_*}}\right) \end{aligned} \tag{4.22}$$

The expression for $q(y_*)$ in (4.20) is analogous to the expression of the approximate predictive distribution of EP-DGP in (3.29). The details of this calculation are shown in Appendix C.3.

As described in Section 3.2.2, there is no analytical solution for the mean of $q(y_*)$, despite $q(y_*)$ can be calculated analytically. Thus, instead of providing the mean, we propose to use the median as an estimator for the targets, as in the case of EP-DGP. Then, the calculation of the predictive median (or other desired quantile) follows the same procedure shown in 3.2.2 for the case of EP-DGP.

4.2 Experiments

We present here the experimental results to evaluate the performance of L-DGP compared with the EP-DGP algorithm and MCMC-DGP. We also include standard GPR, VHGPR, and SVR as comparison benchmarks. First, we assess the quality of the Laplace approximation compared with the other DGP methods on a synthetic data set. Secondly, we present experimental results on several small and medium size data sets, including a computational cost experiment for one of the proposed problems. Finally, we show experimental results on larger data sets.

4.2.1 Synthetic experiment

For the synthetic experiment we have worked with the synthetic heteroscedastic problem proposed in [Yuan and Wahba, 2004] that was previously described in Chapter 3.

According to the proposed mean and noise power distributions, we have generated 200 samples to train the three DGP methods and the standard GPR. For the standard GP, we have used an SE covariance function (see equation (3.37)). For DGP methods, we have used an SE covariance function for $g(\mathbf{x})$ and a SE plus noise covariance function for $f(\mathbf{x})$. As in the case of EP-DGP, assuming that $f(\mathbf{x})$ is a noisy function, it is possible to establish a tradeoff between the heteroscedastic noise component modelled by $c/(g^{+}(\mathbf{x}))^2$ and the homoscedastic noise in $f(\mathbf{x})$.

To initialize EP-DGP and L-DGP hyperparameters we have applied the procedure described in Section 3.3.2 for both methods. As said in that section, we have not implemented any search procedure for MCMC-DGP hyperparameters. Instead, to evaluate the quality of the L-DGP approximation, we have set MCMC-DGP hyperparameters equal to those obtained from the trained L-DGP.

The results for this synthetic experiment are shown in Figure 4.1. For the DGP methods, we show the estimation of quantiles 0.023 and 0.977, that are equivalent to twice the standard deviation in the case of the standard GPR. We can observe that the solutions provided by all DGP methods are very similar, both in the median and the quantiles estimation. Moreover, it is interesting to note that the results of L-DGP and the exact solution provided by MCMC-DGP (using the same set of hyperparameters than L-DGP) almost match, showing the good quality of the proposed Laplace approximation.

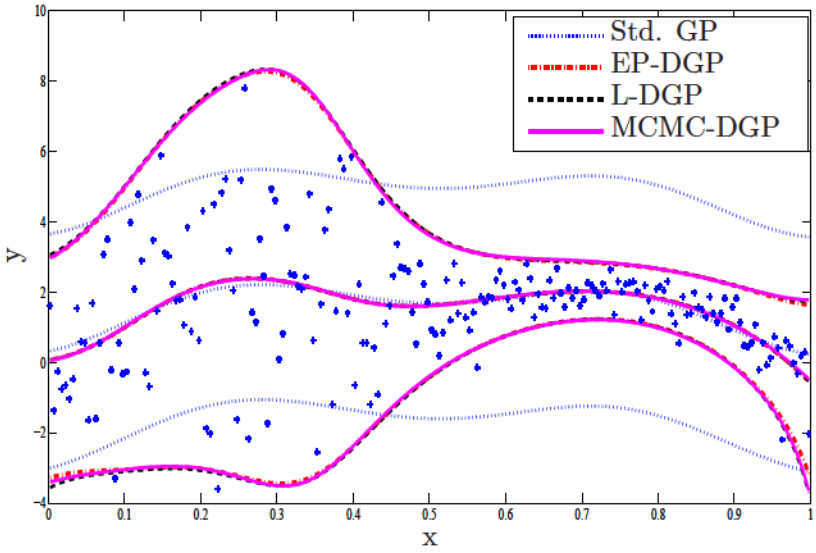


Figure 4.1: Synthetic experiment using data set *Wah* with 200 training samples. The estimated mean and twice the standard deviation are given for the standard GP prediction (dotted line). Median estimation is provided for EP-DGP (dashed-dotted line), L-DGP (dashed line), and MCMC-DGP (continuous line), along with the quantiles 0.023 and 0.977.

With the same data set, we have performed another synthetic experiment to analyse the performance degradation when reducing the number of samples and to compare the results of L-DGP and the standard GP with respect to an optimal

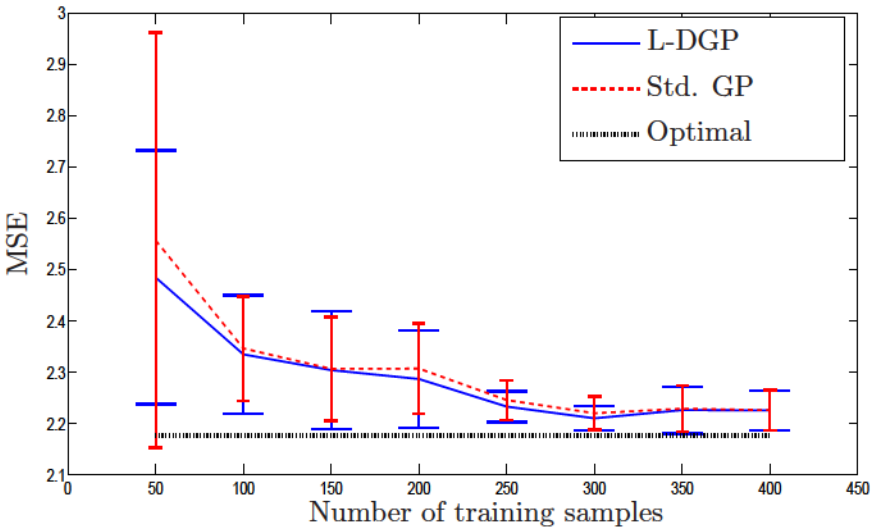
estimator that uses the previously described underlying known distribution. We have explored values of the number of training samples from 50 to 450 with steps of 50 samples. Then, for each size of the training set, we have generated 20 independent data sets from the known distribution. For testing, we have used a unique set of 1000 samples for all the training sets generated for this experiment.

We have used the MSE and the NLPD (see equation (3.40)) as performance measures. The results in terms of MSE are shown in Figure 4.2.(a). The performances of the standard GP and L-DGP are similar, although the L-DGP error is slightly lower in some cases. It can be appreciated that the MSEs of both GP methods are close to the optimal estimator MSE for 300 or more training samples. Concretely, the average difference between L-DGP and the optimal estimator is about 1.6% for 300 training samples.

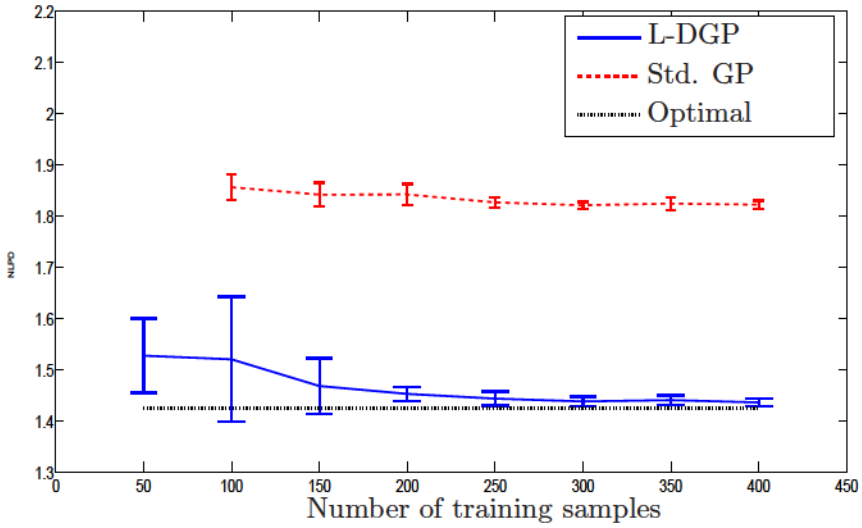
The results of the experiment in terms of NLPD are shown in Figure 4.2.(b). We have omitted the NLPD of the standard GP for 50 training samples for not clutter up the graphic, as a mean value of 4.575 is obtained. Nevertheless, it can be observed a significant difference between the standard GP and L-DGP performances for the rest of the points. This is consistent with the results plotted in Figure 4.1, where the limitations of the standard GP to model heteroscedastic noise are clearly evidenced. It can also be noticed that, as in the case of MSE performance, the NLPD of L-DGP is very close to the NLPD of the optimal estimator for 300 and more training samples. Moreover, the average difference between L-DGP and the optimal estimator is about 1.0% for 300 training samples.

4.2.2 Regression performance

Now, we present experiments to evaluate the performance of L-DGP on several synthetic and real data sets, comparing the results with EP-DGP, MCMC-DGP, the



(a)



(b)

Figure 4.2: MSE (a) and NLPD (b) performance on the synthetic data set for different numbers of training samples. Error bars showing the average MSE, along with the standard deviations, are provided for L-DGP (continuous line and wide error bars) and the standard GP (dashed line and narrow error bars). The results are compared with the optimal estimator built using the known distribution (dotted line).

standard GP, VHGPR, and SVR.

As performance measures we have used the NMSE (3.38), NMAE (3.39), and NLPD (3.40). To train EP-DGP and L-DGP we have initialized the hyperparameters using the same procedure described in Section 3.3.2. As said before, we have not included any hyperparameter search procedure for MCMC-DGP. Instead, in order to assess the quality of the Laplace approximation, we have used the set of hyperparameters found by L-DGP to train the MCMC-DGP method.

For VHGPR, we have used an SE covariance function for the mean latent function and an SE covariance function plus noise for log-noise latent function. To initialize the hyperparameters, we previously train a standard GP following the procedure described in [Lázaro-Gredilla and Titsias, 2011].

For SVR, we have use an RBF kernel. As in the experiments of Chapter 3, the kernel width σ and the cost parameter C have been set to the values that minimize the averaged test NMSE over all the splits created for each data set. Notice, again, that this induces a clear advantage for SVR designs with respect to the other methods, as discussed in Section 3.3.2. The values of C are explored in the range $[2 \cdot 10^{-5}, 2 \cdot 10^{10}]$ with values of the form $2 \cdot 10^{p}$ with $-5 \leq p \leq 10$. For σ, we have also explored values of that form in the range $[2 \cdot 10^{-5}, 2 \cdot 10^{5}]$. As in Section 3.3.2, to calculate the NLPD for SVR predictions we have assumed a Gaussian distribution for the predictions with a constant noise power estimate given by the MSE of the train predictions.

For the experiments we have used 9 data sets: 5 of them (*Wah*, *Ozo*, *Hou*, *Con*, and *Par*) were previously used for the experiments in Chapter 3. The rest of the data sets are the following:

- *Bod* is a data set that tries to estimate the body fat percentage of 252 men using underwater weighing and various body circumference measurements [Penrose et al., 1985].

- In *Ser* data set the rise time of a servomechanism is estimated from gain settings parameters and choices of mechanical linkages [Quinlan, 1992].
- *Aut* data set consists on 7 car attributes to predict the fuel consumption in miles per gallon [Quinlan, 1993].
- *Win* data set tries to estimate the quality of different variants of the Portuguese 'Vinho Verde' red wine from 11 physicochemical measures [Cortez et al., 2009].

The characteristics of the data sets that were previously used in Chapter 3 are shown in Table 3.3, whereas the characteristics of the rest of the data sets are shown in Table 4.1.

Similarly to the experimental procedure described in Chapter 3, for the smallest problems (*Wah*, *Ozo*, *Bod*, *Ser*, *Aut*, *Hou*, and *Con*) we have made 300 random splits to obtain a more complete evaluation of the performance of the compared algorithms. For *Win* and *Par*, we have made a single split due to the high computational costs of the GP methods used in the experiments.

Table 4.1: Characteristics of the data sets used for the experiments.

Data set	Dim	# Tr samples	# Test samples
Bod	13	202	50
Ser	4	134	33
Aut	7	314	78
Win	11	1700	3198

To split the data in training and test sets we have applied the same procedure proposed for the experiments in Chapter 3. Then, for *Wah*, we have generated 300 independent splits with 80 training samples and 20 test samples drawn from the mean and noise distribution given in [Yuan and Wahba, 2004], i.e., with different training

and test samples for each split. For the real data sets *Ozo*, *Bod*, *Ser*, and *Aut* we have made 300 random splits with 80% training samples and 20% test samples. For data sets *Hou* and *Con* we have also made 300 random splits with 50% training and test samples.

The results of the experiments for the small and medium-size data sets are presented in Table 4.2. We do not present results for MCMC-DGP on data set *Con* because the computational burden is very high. We have performed a Wilcoxon Rank-Sum test to assess statistical significance of the differences at the 5% level.

From the results in Table 4.2, it can be seen that SVR performs worse than the GP based methods with statistical significant differences in 5 of the 7 problems in terms of NMSE, NMAE, and NLPD. SVR only performs better than GP methods in data set *Wah* with respect to NMSE and NMAE measures, with significant differences only with respect to the standard GP. It can also be noticed that the three DGP methods never perform worse than the standard GP in any of the three performance measures. Even when the problem is homoscedastic, as it seems to be the case for data set *Bod*, the performances of the DGP methods are the same than the performance of the standard GP.

In terms of NMSE, we can say the following:

- L-DGP, MCMC-DGP, and EP-DGP performances are similar. However there are statistical significant improvements of L-DGP with respect to EP-DGP in *Con* and with respect to MCMC-DGP and EP-DGP in *Hou*.

- DGP methods also outperform VHGPR with significant differences in 4 of the 7 data sets.

- L-DGP improves the standard GP results in *Ozo*, *Ser*, and *Con*, whereas MCMC-DGP and EP-DGP outperform the standard GP in *Ozo* and *Con*.

Table 4.2: Experimental test results on small and medium size data sets. Average NMSE, NMAE, and NLPD plus/minus one standard deviation. Statistically significant improvements are marked as • w.r.t. standard GP, ○ w.r.t. VHGPR, ⋆ w.r.t. EPDGP, ◇ w.r.t. LDGP, ▷ w.r.t. MCMC-DGP, and † w.r.t. SVR. Statistical significance is measured according to a Wilcoxon Rank-Sum test at the 5% level.

		Average NMSE	Average NMAE	Average NLPD
Wah:	Std. GP	0.956 ± 0.179	0.951 ± 0.096	1.911 ± 0.311
	VHGPR	0.937 ± 0.146	0.941 ± 0.090	1.600 ± 0.287 • †
	EP-DGP	0.940 ± 0.208	0.938 ± 0.114	1.542 ± 0.274 • ○†
	L-DGP	0.943 ± 0.207	0.941 ± 0.113	1.542 ± 0.269 • ○†
	MCMC-DGP	0.943 ± 0.207	0.941 ± 0.113	1.544 ± 0.272 • ○†
	SVR ($C = 200; \sigma = 0.2$)	0.908 ± 0.190 •	0.915 ± 0.109 •	1.893 ± 0.340
Ozo:	Std. GP	0.281 ± 0.091 †	0.484 ± 0.085	4.323 ± 0.271 †
	VHGPR	0.282 ± 0.083 †	0.477 ± 0.079	4.156 ± 0.209 • †
	EP-DGP	0.258 ± 0.079 • ○†	0.462 ± 0.079 • ○†	4.071 ± 0.151 • ○†
	L-DGP	0.256 ± 0.079 • ○†	0.460 ± 0.079 • ○†	4.072 ± 0.137 • ○†
	MCMC-DGP	0.257 ± 0.080 • ○†	0.461 ± 0.079 • ○†	4.073 ± 0.143 • ○†
	SVR ($C = 200; \sigma = 0.2$)	0.300 ± 0.094	0.486 ± 0.085	4.443 ± 0.530
Bod:	Std. GP	0.290 ± 0.061	0.525 ± 0.058	-0.989 ± 0.096 †
	VHGPR	0.290 ± 0.061	0.525 ± 0.058	-0.989 ± 0.097 †
	EP-DGP	0.291 ± 0.061	0.526 ± 0.058	-0.988 ± 0.098
	L-DGP	0.290 ± 0.061	0.525 ± 0.058	-0.989 ± 0.096 †
	MCMC-DGP	0.291 ± 0.061	0.526 ± 0.058	-0.988 ± 0.098
	SVR ($C = 20; \sigma = 0.002$)	0.294 ± 0.055	0.533 ± 0.057	-0.970 ± 0.116
Ser:	Std. GP	0.166 ± 0.090 †	0.302 ± 0.062 †	-0.871 ± 0.676 †
	VHGPR	0.182 ± 0.124 †	0.267 ± 0.076 • †	-1.972 ± 0.412 • ⋆ ◇ ▷†
	EP-DGP	0.151 ± 0.109 ○ †	0.245 ± 0.065 • ○†	-1.892 ± 0.173 • †
	L-DGP	0.143 ± 0.104 • ○†	0.227 ± 0.062 • ○ ⋆ ▷†	-1.940 ± 0.138 • ⋆ ▷ †
	MCMC-DGP	0.151 ± 0.109 ○ †	0.245 ± 0.065 • ○†	-1.892 ± 0.173 • †
	SVR ($C = 200; \sigma = 0.02$)	0.224 ± 0.094	0.443 ± 0.070	-0.688 ± 0.650
Aut:	Std. GP	0.116 ± 0.030 †	0.289 ± 0.031 †	-1.234 ± 0.137 †
	VHGPR	0.117 ± 0.029 †	0.288 ± 0.031 †	-1.361 ± 0.117 • †
	EP-DGP	0.119 ± 0.031 †	0.287 ± 0.033 †	-1.354 ± 0.103 • †
	L-DGP	0.116 ± 0.030 †	0.284 ± 0.032 †	-1.359 ± 0.093 • †
	MCMC-DGP	0.119 ± 0.031 †	0.287 ± 0.033 †	-1.354 ± 0.103 • †
	SVR ($C = 2; \sigma = 0.02$)	0.134 ± 0.028	0.321 ± 0.035	-1.143 ± 0.137
Hou:	Std. GP	0.152 ± 0.035 ○ ◇ ▷ †	0.352 ± 0.022 †	2.624 ± 0.122 †
	VHGPR	0.166 ± 0.034 †	0.352 ± 0.022 †	2.564 ± 0.150 • †
	EP-DGP	0.159 ± 0.037 ○ †	0.345 ± 0.023 • ○†	2.445 ± 0.075 • ○†
	L-DGP	0.152 ± 0.036 ○ ⋆ ▷ †	0.336 ± 0.023 • ○ ⋆ ▷†	2.413 ± 0.057 • ○ ⋆ ▷†
	MCMC-DGP	0.159 ± 0.037 ○ †	0.345 ± 0.023 • ○†	2.445 ± 0.075 • ○†
	SVR ($C = 200; \sigma = 0.02$)	0.177 ± 0.042	0.356 ± 0.024	3.126 ± 0.500
Con:	Std. GP	0.132 ± 0.014 †	0.326 ± 0.014 †	3.162 ± 0.041 †
	VHGPR	0.134 ± 0.013 †	0.327 ± 0.014 †	3.088 ± 0.042 • †
	EP-DGP	0.123 ± 0.015 • ○†	0.310 ± 0.014 • ○†	3.049 ± 0.041 • ○ ◇ †
	L-DGP	0.120 ± 0.014 • ○ ⋆ †	0.304 ± 0.014 • ○ ⋆ †	3.060 ± 0.036 • ○ †
	SVR ($C = 2000; \sigma = 0.02$)	0.158 ± 0.017	0.353 ± 0.016	3.391 ± 0.106

The results in terms of NMAE show that:

- L-DGP, MCMC-DGP, and EP-DGP perform similarly except for *Hou*, where L-DGP outperforms MCMC-DGP and EP-DGP, and *Con*, where L-DGP improves EP-DGP performance.
- DGP methods improve with statistical significance GP and VHGPR in *Ozo*, *Ser*, *Hou*, and *Con*.
- In none of the data sets DGP algorithms perform worse than the other GP methods.

Finally, in terms of NLPD it can be observed that:

- VHGPR and the DGP algorithms outperform the standard GP in 6 of the 7 proposed data sets with statistical significance, and are not worse for *Bod*.
- Comparing the differences between the DGP methods and VHGPR, it can be appreciated that DGP methods outperform VHGPR in 4 data sets, whereas VHGPR only outperforms DGP methods in *Ser*.
- The performance of L-DGP, MCMC-DGP, and EP-DGP is similar, although L-DGP improves NLPD perofmance in *Ser* and *Hou*, whereas EP-DGP outperforms EP-DGP in *Con*.

The experimental results for the biggest data sets (*Win* and *Par*) are shown in Table 4.3. For data set *Win* we observe that:

- The performance in terms of NMSE and NMAE is similar for all the GP methods.
- EP-DGP and L-DGP have better results in terms of NLPD with respect to the standard GP and VHGPR.

Table 4.3: Experimental test results on data sets *Win* and *Par*. NMSE, NMAE, and NLPD are provided for the standard GP, VHGPR, EP-DGP, L-DGP, and SVR.

		NMSE	NMAE	NLPD
Win:	Std. GP	0.653	0.838	1.059
	VHGPR	0.657	0.842	1.057
	EP-DGP	0.652	0.839	1.014
	L-DGP	0.649	0.837	1.018
	SVR ($C = 2; \sigma = 0.2$)	0.667	0.818	1.267
Par:	Std. GP	0.216	0.396	−1.983
	VHGPR	0.222	0.395	−2.060
	EP-DGP	0.170	0.370	−2.063
	L-DGP	0.219	0.394	−2.049
	SVR ($C = 200; \sigma = 2 \cdot 10^{-4}$)	0.231	0.487	−1.700

- SVR outperform GP methods in terms of NMAE. However, NMSE and NLPD performance is worse.

For data set *Par* we note that:

- EP-DGP outperforms the other GP methods and SVR in terms of NMSE and NMAE.
- NLPD performance is similar for EP-DGP and VHGPR, with a slight advantage with respect to L-DGP.
- SVR also performs clearly worse than the other GP methods in terms of NLPD.

The results of these experiments, as in the case of the synthetic experiment, support the good quality of the Laplace approximation when comparing it with the

exact posterior provided by MCMC-DGP using the same set of hyperparameters. Moreover, the performance of L-DGP is similar to the performance of EP-DGP. However, the computational cost of the proposed Laplace approximation is lower and the convergence is assured, whereas EP-DGP is conjectured but not proven to converge, even when the likelihood is log-concave.

4.2.3 Time scalability experiment

The complexity of all the GP methods used in the experiments scales in the form $\mathcal{O}(N^3)$. However, the training time of these methods can be quite different. To illustrate this point, we have performed an experiment with data set *Hou* to measure the training time for the standard GP, VHGPR, EP-DGP, and L-DGP with different number of training samples. Hence, we have measured the training time avoiding the hyperparameters learning varying the number of training samples from 50 to 500. For a given number of training samples, we have generated 20 random training sets. To set the hyperparameters we have first trained all the GP methods using all the available samples as training samples.

The experiments have been conducted in a 4 GB computer with an Intel core i5 processor at 3.33 GHz using Matlab implementations for all the GP algorithms. The average training time along with the corresponding error bars are shown in Figure 4.3. First of all, it can be appreciated that all the GP methods scale similarly with the number of samples, with is consistent with the theoretic $\mathcal{O}(N^3)$ time scalability mentioned before. Despite the standard GP is the fastest method, it can be noticed that the time required to train L-DGP is lower than the VHGPR and EP-DGP training times. The training time reduction of L-DGP with respect to EP-DGP is remarkable. For example, for 500 training points, EP-DGP approximately takes 2000 seconds to train the model, whereas L-DGP takes only 200 seconds (10 times faster).

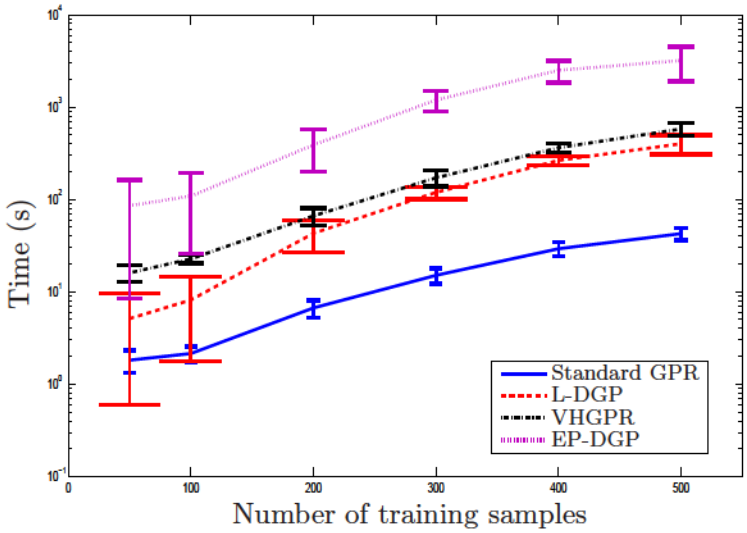


Figure 4.3: Results of the time experiment using data set *Hou* with different training sizes. Error bars are shown to represent the training time for the standard GP (continuous line and narrower error bars), L-DGP (dashed line and wider error bars), VHGPR (dashed-dotted line), and EP-DGP (dotted line).

4.3 Conclusions

The high computational burden of the EP-DGP method described in Chapter 3 to perform inference on the DGP and the lack of a formal proof of EP convergence (even when the likelihood is log-concave) motivate the use of the Laplace approximation to perform inference in the DGP model. The likelihood log-concavity ensures a unimodal posterior which allows the Laplace approximation to converge to a unique maximum.

Since the likelihood of the DGP model has a quite Gaussian shape, a good posterior approximation of L-DGP is expected. The experimental comparisons with the exact posterior sampled with MCMC-DGP corroborate this statement, showing that L-DGP also performs similarly to EP-DGP.

As in Chapter 3, the experimental results confirm the advantages of DGP methods to perform nonstationary regression compared to the standard GP, SVR, and the state-of-the-art in heteroscedastic regression VHGPR. The computational cost experiment evidences that the time required to train L-DGP is considerably lower than the EP-DGP training time. Moreover, the results show also a lower computational burden of L-DGP with respect to VHGPR in the considered problem.

Consequently, the similar regression performance compared to EP-DGP, the reduced computational burden (despite the $\mathcal{O}(N^3)$ scalability), and the convergence guarantee of the algorithm make L-DGP a good alternative to make inference for the DGP model to achieve nonstationary and heteroscedastic regression.

Chapter 5

Laplace approximation with Gaussian processes for volatility forecasting

In this chapter we present an heteroscedastic GP method using the Laplace approximation for volatility forecasting in financial time series. Volatility prediction is a critical task for asset valuation and risk management in financial markets for investors and financial intermediaries [Brownlees et al., 2011]. A good forecast of the volatility of asset prices over the investment holding period can be useful for investment purposes [Poon and Granger, 2003]. The importance of volatility forecasting was highlighted in 2003 when Robert Engle received the Nobel prize in Economics for his outstanding research in modelling volatility dynamics.

Volatility is defined as the standard deviation of a return series at time instant[1] x given the information available at time instant $x-1$. Denoting the price or assets

[1] We will use x to denote time to be consistent with traditional notation in machine learning for the inputs of a learning algorithm.

series as $P(x)$, the corresponding return series is calculated as

$$y(x) = \log\left(P(x)\right) - \log\left(P(x-1)\right) \tag{5.1}$$

Then, the volatility corresponds to the standard deviation of the noise in the data set constituted by the pairs $\{x, y(x)\}$. The return series can be considered as a zero-mean process. Another usual assumption is that time instants are discrete equally spaced values, such as hours or days. An extensive survey on volatility forecasting methods can be found in [Broto and Ruiz, 2004].

AutoRegressive Conditional Heteroscedastic (ARCH) [Engle, 1983] and Generalized ARCH (GARCH) models [Bollerslev, 1986] are commonly used for volatility preditction. Concretely, as reported in [Hansen and Lunde, 2005], the performance of GARCH(1,1), equivalent to an AutoRegressive Moving Average (ARMA) (1,1) model, has been shown to be very competitive for this task. However, the parameters of these methods are estimated by Maximum Likelihood (ML) given the training data, which makes them prone to overfitting, specially if the number of training samples is small. Then, GP methods can be useful to overcome this limitation, as they have been proven to be resilient to overfitting problems. This is the case of the GP volatility forecasting methods proposed in the literature: In [Girolami and Calderhead, 2011], an efficient MCMC method is proposed to make inference in a Bayesian volatility model in which inference is not analytically tractable. A variational approximation is proposed in [Lázaro-Gredilla and Titsias, 2011], where the VHGPR method, described in Chapter 2, can be easily adapted to perform approximate inference in the same volatility model. On the other hand, [Wilson and Ghahramani, 2010] proposes a Copula GP (CGP) model to predict volatility, where the copula process is a model describing the dependencies between arbitrarily many random variables independently of their marginal distributions.

We propose to use the Laplace method to make approximate inference in the GP volatility model used by the Riemann Manifold Hamiltonian

Monte Carlo (RMHMC) method in [Girolami and Calderhead, 2011] and VHGPR [Lázaro-Gredilla and Titsias, 2011]. Although high quality posterior approximations are obtained with VHGPR at a reduced cost (compared with RMHMC), the Laplace approximation can produce similar posterior approximations compared with VHGPR and RMHMC reducing the computational burden with respect to both methods.

The rest of the chapter is organized as follows: In Section 5.1 we briefly describe the GARCH models, which have been widely used for volatility forecasting during the last years. In Section 5.2 we present the heteroscedastic GP model for volatility forecasting. Then, in Section 5.3 we present the Laplace method for GP Volatility Forecasting (LGPVF). Experimental results on synthetic and real financial data sets are shown in Section 5.4. Finally, we summarize the chapter in Section 5.5.

5.1 GARCH models

GARCH models are commonly used to predict volatility as they are relatively easy to train and provide a very competitive performance. They were proposed by Tim Bollerslev in [Bollerslev, 1986] as an extension of ARCH models.

GARCH models assume that the return series in (5.1) is a zero mean process which is corrupted with input-dependent (heteroscedastic) Gaussian noise, i.e.,

$$y(x) \sim \mathcal{N}(y|0, r(x)) \tag{5.2}$$

where $r(x)$ is the function that models the variance of the heteroscedastic noise. If an ARMA model is assumed for $r(x)$, then, that ARMA model is named GARCH model.

The notation GARCH(p, q) refers to the GARCH model with p autoregressive

terms and q moving average terms, so that

$$r(x) = a_0 + \sum_{i=1}^{p} a_i y^2(x-i) + \sum_{i=1}^{q} b_i \, r(x-i) \tag{5.3}$$

where $a_0 > 0$, $a_i \geq 0$, and $b_i \geq 0$ to ensure that the estimation of the volatility $r(x)$ is positive.

As mentioned before, the parameters of the GARCH models are estimated by constrained ML given the training data. Although this is an easy way to adjust the parameters, ML estimation is prone to overfitting, limiting the generalization capabilities of GARCH to predict future values of the volatility.

For the experiments in Section 5.4 we will use GARCH(1,1), as it has been reported to have very good performance to predict volatility [Hansen and Lunde, 2005].

5.2 GP volatility model

In this section we describe the GP model for volatility forecasting that was previously proposed in [Girolami and Calderhead, 2011] and used later in [Lázaro-Gredilla and Titsias, 2011].

From the regression model used by the standard GPR, where the observations $y(x)$ are modeled as a latent function dependent on the inputs corrupted by Gaussian noise, i.e.,

$$y(x) = f(x) + \varepsilon_n \tag{5.4}$$

if we assume that the latent function f is a zero-mean process and that the Gaussian noise is heteroscedastic, we obtain the same model than GARCH. Then, we can model ε as an input-dependent Gaussian noise process that can be modeled as $\varepsilon \sim \mathcal{N}(\varepsilon|0, e^{g(x)})$. We adopt the exponential of the latent function g to ensure that the noise power estimates are positive. Therefore, the latent function g describes

the logarithm of the input-dependent noise power of the return series. Then, the likelihood of $g(x_n)$ given an observation $y(x_n)$ can be written as

$$p(y_n|g_n) = \mathcal{N}(y_n|0, e^{g_n}) \tag{5.5}$$

where $y_n = y(x_n)$ and $g_n = g(x_n)$. This likelihood corresponds with the Gaussian noise model assumed by GARCH, described in Section 5.1, and the likelihood of the heteroscedastic model in (2.43) used by VHGPR, if we assume that the latent function f is zero.

Following a Bayesian treatment, we place a GP prior on the latent function g, so that

$$g(x) \sim \mathcal{GP}(\mu_0, k_g(x, x')) \tag{5.6}$$

where μ_0 is a mean log-noise hyperparameter. For $k_g(x, x')$ we use a reparameterization of the standard Ornstein-Uhlenbeck covariance function, given by

$$k_g(x, x') = \frac{\sigma_0^2}{1 - \phi^2}\phi^{|x-x'|} \tag{5.7}$$

where σ_0 and ϕ are the covariance function hyperparameters.

This covariance function is a special case of the Matérn class of covariance functions [Rasmussen and Williams, 2006] that corresponds to the Ornstein-Uhlenbeck process [Uhlenbeck and Ornstein, 1930], which was introduced as a mathematical model of the velocity of a particle undergoing Brownian motion, so that it is also appropriate to model the behavior of the noise in the return series. In this case, it reduces $g(x)$ to an AR(1) process when the inputs x, x', are restricted to be integer values. This corresponds to the volatility model proposed in [Liu, 2001]. Moreover, if the inputs are ordered and equally spaced, as in the volatility scenario described in Section 5.2, $g(x)$ is a first-order Markov process.

Under this assumption, although the resulting covariance matrix K_g is a full

matrix, its inverse K_g^{-1} results tridiagonal:

$$K_g^{-1} = \begin{pmatrix} 1/\sigma_0^2 & -\phi/\sigma_0^2 & & & 0 \\ -\phi/\sigma_0^2 & \frac{1+\phi^2}{\sigma_0^2} & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \frac{1+\phi^2}{\sigma_0^2} & -\phi/\sigma_0^2 \\ 0 & & & -\phi/\sigma_0^2 & 1/\sigma_0^2 \end{pmatrix} \tag{5.8}$$

As the likelihood in (5.5) is not Gaussian on g_n,, analytical inference of the posterior, the evidence, and the predictive distribution are intractable. Hence, as explained in Chapter 2, to make inference in the model we need to apply MCMC procedures or approximate inference techniques. In [Girolami and Calderhead, 2011] RMHMC is proposed to efficiently drawn samples from the posterior distribution integrating over the hyperparameters of the GP prior. Although the computational cost is high, compared to approximate inference methods, RMHMC exploits the tridiagonal character of K_g^{-1} to increase the efficiency of the algorithm, so that it can be applied to model volatility for relatively large data sets. On the other hand, as said before, VHGPR can be easily adapted to perform inference on this volatility model. Taking advantage of the tridiagonal property of K_g^{-1} it is possible to make the implementation of the algorithm scaling in linear time, instead of the $\mathcal{O}(N^3)$ computation time required for other GP methods. For the case of the LGPVF algorithm, that we describe next, it is also possible to obtain a $\mathcal{O}(N)$ implementation using this particularity of the inverse covariance matrix.

5.3 Inference with LGPVF

The likelihood of the GP volatility model (5.5) is log-concave. As explained in the previous chapters, this leads to a unimodal posterior, which favors the convergence of EP approximations (it is conjectured to converge but not proven, as was discussed

in Chapter 2) and makes that the Laplace method converges to a unique global maximum. Moreover, as the likelihood has a close-to-Gaussian density on g_n, the true posterior should be nearly Gaussian. Then, the Laplace method is appealing and accurate posterior approximations can be expected, similarly to other approximate inference algorithms, such as EP or Variational Bayes, but at a reduced cost.

To simplify the calculations with LGPVF, we have used an equivalent volatility model to that presented in Section 5.4, moving the GP prior offset μ_0 in (5.6) to the likelihood, so that the modified GP prior can be written as

$$g(x) \sim \mathcal{GP}(0, k_g(x, x')) \tag{5.9}$$

and the modified likelihood becomes

$$p(y_n|g_n) = \mathcal{N}\left(y_n|0, e^{(g_n+\mu_0)}\right) \tag{5.10}$$

As explained in Chapter 2, the Laplace method approximates the real posterior[2] $p(\mathbf{g}|\mathbf{y})$ with a Gaussian distribution $q(\mathbf{g}|\mathbf{y})$ doing a second order Taylor expansion of $\log p(\mathbf{g}|\mathbf{y})$ around the maximum of the posterior

$$q(\mathbf{g}|\mathbf{y}) = \mathcal{N}(\mathbf{g}|\hat{\mathbf{g}}, A^{-1}) \propto \exp\left(-\frac{1}{2}(\mathbf{g}-\hat{\mathbf{g}})^T A(\mathbf{g}-\hat{\mathbf{g}})\right) \tag{5.11}$$

where $\hat{\mathbf{g}} = \arg\max_{\mathbf{g}} p(\mathbf{g}|\mathbf{y})$ and $A = -\nabla\nabla \log p(\mathbf{g}|\mathbf{y})|_{\mathbf{g}=\hat{\mathbf{g}}}$ is the Hessian of the negative log-posterior at $\mathbf{g} = \hat{\mathbf{g}}$.

Maximizing the log-posterior is equivalent to maximizing the functional:

$$\Psi(\mathbf{g}) = \log p(\mathbf{y}|\mathbf{g}) + \log p(\mathbf{g}) = \log p(\mathbf{y}|\mathbf{g}) - \frac{1}{2}\mathbf{g}^T K_g^{-1}\mathbf{g} - \frac{1}{2}\log|K_g| - \frac{N}{2}\log 2\pi \tag{5.12}$$

Then, making $\nabla\Psi(\mathbf{g}) = 0$ we obtain the following self-consistent equation at the maximum of $\Psi(\mathbf{g})$ that can be solved applying the Newton method, whose update equation is given by

$$\hat{\mathbf{g}} = K_g(\nabla \log p(\mathbf{y}|\hat{\mathbf{g}})) \tag{5.13}$$

[2]For the sake of clarity we omit conditioning on the inputs $\mathbf{X}$ and the set of hyperparameters $\boldsymbol{\theta}$.

where the maximum, as described in Chapter 2, can be searched using the Newton method. The update equation of the Newton method is:

$$\mathbf{g}_{new} = (K_g^{-1} + W)^{-1}(W\mathbf{g} + \nabla \log p(\mathbf{y}|\mathbf{g})) \tag{5.14}$$

where $\nabla \log p(\mathbf{y}|\mathbf{g})$ is a vector of length N with

$$[\nabla \log p(\mathbf{y}|\mathbf{g})]_n = \frac{y_n^2}{2e^{(g_n+\mu_0)}} - 1/2 \tag{5.15}$$

and $W = -\nabla\nabla \log p(\mathbf{y}|\mathbf{g})$ is the negative Hessian of the log-likelihood. As this likelihood is log-concave, the Hessian results in a diagonal negative definite matrix, so that the diagonal elements of W have to be positive, with $[W]_{n,n} = y_n^2/\left(2e^{(g_n+\mu_0)}\right)$. This leads $\Psi(\mathbf{g})$ to be concave, so that it is guaranteed to have a unique maximum.

The tridiagonal character of K_g^{-1} makes possible to compute (5.14) in $\mathcal{O}(N)$.

Then, the Gaussian posterior approximation provided by the Laplace method can be written as

$$q(\mathbf{g}|\mathbf{y}) = \mathcal{N}(\hat{\mathbf{g}}, (K_g^{-1} + W)^{-1}) \tag{5.16}$$

The approximate evidence needed to search the hyperparameters with an ML-II implementation is given by

$$\log q(\mathbf{y}) = -\frac{1}{2}\hat{\mathbf{g}}^T K_g^{-1}\hat{\mathbf{g}} + \log p(\mathbf{y}|\hat{\mathbf{g}}) - \frac{1}{2}\log(|B|) \tag{5.17}$$

with

$$|B| = |K_g| \cdot |K_g^{-1} + W| = \frac{|K_g^{-1} + W|}{|K_g^{-1}|} \tag{5.18}$$

where both determinants can be computed in $O(N)$ time as the matrices are tridiagonal.

The derivatives of the log-evidence with respect to the hyperparameters can be found in Appendix E.

The approximate predictive distribution for new samples can be obtained as

$$q(g_*|\mathbf{y}) = \int p(g_*|\mathbf{g},\mathbf{y})q(\mathbf{g}|\mathbf{y})d\mathbf{g} = \mathcal{N}(g_*|\mu_*\sigma_*^2) \tag{5.19}$$

with

$$\mu_* = \mathbf{k}_{g_{*N}}^T(\nabla \log p(\mathbf{y}|\hat{\mathbf{g}})) \tag{5.20}$$

$$\sigma_*^2 = k_{g_{**}} - \mathbf{k}_{g_{*N}}^T(K_g + W^{-1})^{-1}\mathbf{k}_{g_{*N}} \tag{5.21}$$

To allow the implementation of (5.21) scaling linearly with the number of samples we have to apply the matrix inversion lemma (see Appendix B.1 for details) to the term $(K_g + W^{-1})^{-1}$, so that

$$(K_g + W^{-1})^{-1} = K_g^{-1} - K_g^{-1}(K_g^{-1} + W)^{-1}K_g^{-1} \tag{5.22}$$

5.4 Experiments

5.4.1 A synthetic experiment

To assess the quality of LGPVF, we have reproduced the synthetic volatility example presented in [Girolami and Calderhead, 2011] to assess the quality of RMHMC. In this method, the hyperparameters σ_0, ϕ, and $\beta = \exp(\mu_0/2)$ are integrated over, so they do not need to be learned. To reduce the high computational burden of traditional MCMC techniques, RMHMC also exploits the tridiagonal property of K_g^{-1} to make the algorithm more efficient, so that it can be used for relatively large data sets.

In the experiment proposed in [Girolami and Calderhead, 2011], 2000 data points are generated from the volatility model in (5.5), using parameters $\sigma_0 = 0.15$, $\phi = 0.98$, and $\beta = 0.65$. With this data set, we compute the approximate posterior using LGPVF and VHGPR, learning the hyperparameters with a standard ML-II implementation. The hyperparameters are initialized as proposed in

[Lázaro-Gredilla and Titsias, 2011] and [Girolami and Calderhead, 2011]: $\sigma_0 = 0.50$, $\phi = 0.50$, and $\beta = 0.50$.

Under these settings, the results of the mean of the posterior over the hyperparameters obtained by RMHMC and the values of the hyperparameters learned by LGPVF and VHGPR are shown in Table 5.1. It can be appreciated that the hyperparameters obtained by VHGPR and LGPVF are very close to the results of RMHMC and to the ground truth values. These results support the accuracy of the proposed Laplace approximation.

Table 5.1: Values of the hyperparameters obtained by VHGPR and LGPVF, and mean of the posterior hyperparameters obtained by RMHMC on the synthetic data set. Ground truth values are also shown to highlight the accuracy of the three compared methods.

Method	σ_0	ϕ	β
RMHMC	0.171	0.977	0.665
VHGPR	0.148	0.981	0.666
LGPVF	0.152	0.980	0.665
Ground truth	0.150	0.980	0.650

Moreover, the posterior over $g(x)$ produced by RMHMC and LGPVF is plotted in Figure 5.1. It can be observed that the means of both posteriors are very similar, although the predictive variance of RMHMC is slightly higher. This is due to the effect of integrating over the hyperparameters in the RMHMC method, as it is also shown in [Lázaro-Gredilla and Titsias, 2011]. For the sake of clarity, we have omitted the comparison with the posterior of VHGPR in Figure 5.1, as it almost matches the solution of LGPVF.

The results of this synthetic experiment show that the approximate posterior

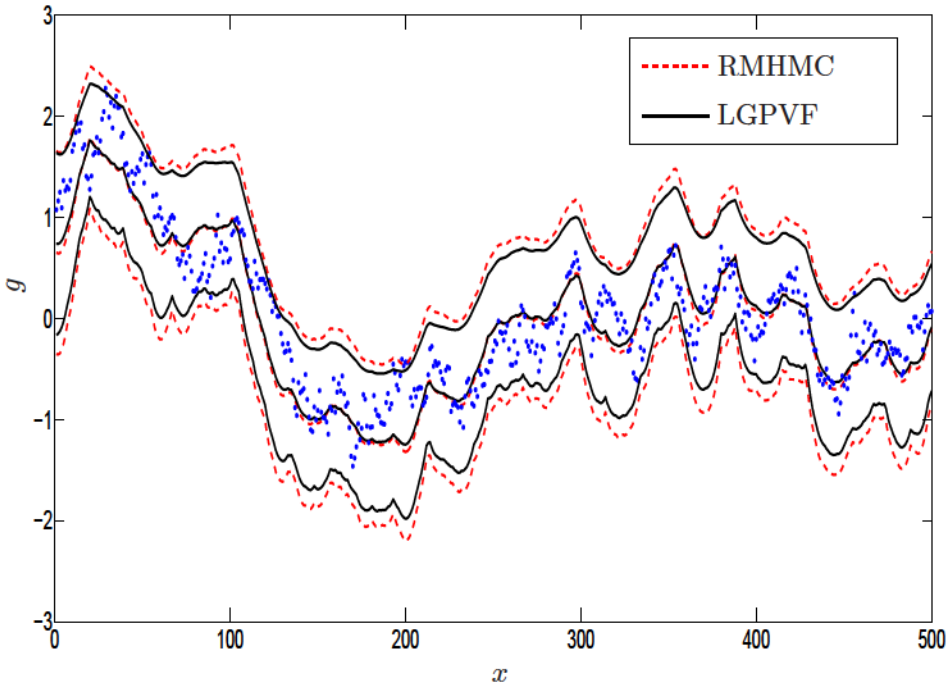


Figure 5.1: Posterior over g for the synthetic volatility data set using RMHMC (dashed line) and LGPVF (continuous line). The mean and twice the standard deviation error bars are shown for both methods. For clarity, only the first 500 points are shown.

of LGPVF is very close to the (asymptotically exact) posterior given by RMHMC. However, the computational cost of the Laplace method is much lower than the burden of the MCMC method, despite RMHMC also exploits the tridiagonal character of the inverse covariance matrix to speed-up inference.

5.4.2 Experiments with financial time series

For the second part of the experiments, we have used real world financial data sets to compare LGPVF with VHGPR and the commonly used GARCH (1,1) for volatility prediction tasks, which provides satisfactory results in many cases.

We have considered 5 return series of the daily exchange rate between different currencies: *Euro - American Dollar*, *Euro - Yuan*, *Great Britain Pound (GBP) - American Dollar*, *Euro - Rupee*, and *Yuan - Rupee* [3]. We have also included another standard financial data set in volatility forecasting [Wilson and Ghahramani, 2010, McCullough and Renfro, 1998, Brooks et al., 2001] consisting on the return series of the daily exchange rate between the Deutsch Mark (DEM) and the GBP. The characteristics of these data sets appear in Table 5.2.

We have used the implementation provided by Matlab[4] for GARCH(1,1). To set the hyperparameters of the GP models, we have used the first 90 days for training, giving as initial values for the hyperparameters: $\sigma_0 = 1$, $\phi = 0.9$, and $\beta = 10^{-3}$. After this first stage, we have made 1, 7, 30, and 60 days ahead volatility forecasts each trading day, retraining the models every 7 days using a rolling window with the previous 30 days. Predictions have been made for all the trading days except the first 90 days.

As a performance measure, we have used the MSE between the predicted volatility

[3]The data has been collected from http://www.oanda.com.

[4]We have used Matlab version 7.10.0 R2010a.

Table 5.2: Characteristics of the currencies exchange rate data sets used in the experiments.

Currencies	Period	Total trading days
DEM - GBP	January 1984 - January 1992	1974
Euro - Dollar	January 31st 2009 - January 31st 2014	1826
Euro - Yuan	January 31st 2009 - January 31st 2014	1826
GBP - Dollar	January 31st 2009 - January 31st 2014	1826
Euro - Rupee	January 31st 2009 - January 31st 2014	1826
Yuan - Rupee	January 31st 2009 - January 31st 2014	1826

and the squared returns, as suggested in [Brownlees et al., 2011]. Then, we have averaged the MSE for all the trading days for which forecasts are calculated.

We also have measured the averaged training time for the three methods, not considering the initial training time for the GP methods, as the GARCH model does not need to be previously trained to initialize its parameters, so that we only take into account the time to (re)train the models each 7 trading days[5].

The results of the experiments shown in Table 5.3 show a better performance of both GP methods when compared to GARCH(1,1) in all data sets, mainly for 30 and 60 day forecasts. It is also interesting to observe that the predictions of the GP algorithms slightly degrade when increasing the forecast horizon, whereas this degradation is important in the case of GARCH(1,1). On the other hand, the performance of VHGPR and LGPVF is very similar in most cases, with very slight MSE improvements of LGPVF with respect to VHGPR in some cases, though they cannot be considered significant. However, the time required to retrain LGPVF

[5]The experiments have been conducted on a 8 GB computer with an Intel core i7-3610Q at 2.30 GHz using a Matlab implementation.

Table 5.3: Averaged MSE for different forecast horizons and averaged (re)training times using GARCH(1,1), VHGPR, and LGPVF.

Data set	Method	MSE ($\times 10^{-9}$) - Days ahead				Avg. tr. time
		1	7	30	60	(s)
DEM-GBP	GARCH (1,1)	2.946	3.431	9.855	28.634	0.437 ± 0.186
	VHGPR	2.902	2.984	3.145	3.121	0.120 ± 0.060
	LGPVF	2.902	2.984	3.145	3.122	0.065 ± 0.022
Euro-Dollar	GARCH (1,1)	1.160	1.272	2.298	5.092	0.444 ± 0.191
	VHGPR	1.112	1.117	1.125	1.140	0.108 ± 0.065
	LGPVF	1.111	1.116	1.119	1.137	0.069 ± 0.026
Euro-Yuan	GARCH (1,1)	1.222	1.297	2.154	4.411	0.432 ± 0.189
	VHGPR	1.079	1.084	1.103	1.122	0.107 ± 0.068
	LGPVF	1.078	1.084	1.097	1.121	0.066 ± 0.029
GBP-Dollar	GARCH (1,1)	0.681	0.688	1.094	2.373	0.452 ± 0.188
	VHGPR	0.663	0.635	0.622	0.618	0.119 ± 0.078
	LGPVF	0.668	0.636	0.620	0.618	0.064 ± 0.022
Euro-Rupee	GARCH (1,1)	3.776	4.116	6.385	12.679	0.438 ± 0.193
	VHGPR	3.636	3.711	3.771	3.733	0.112 ± 0.068
	LGPVF	3.636	3.711	3.771	3.733	0.059 ± 0.017
Yuan-Rupee	GARCH (1,1)	3.654	3.949	4.194	4.747	0.446 ± 0.179
	VHGPR	3.927	4.005	4.042	4.091	0.115 ± 0.072
	LGPVF	3.927	4.005	4.042	4.091	0.064 ± 0.019

is sensibly lower than the retraining time of VHGPR, reducing the computational burden in around 40%. This makes the Laplace approximation more appealing, since a similar error performance is obtained at a reduced cost. It is also very significant that LGPVF is trained approximately 7 times faster than GARCH(1,1) in all data

sets.

In the case of the DEM-GBP data set, the results are consistent with those presented in [Lázaro-Gredilla and Titsias, 2011] and [Wilson and Ghahramani, 2010]. Although in these two papers the rolling window to retrain the models is set to 120 days, whereas we are using only a 30 day window, the MSE performance is similar. This suggests that increasing the number of training samples does not imply a significantly better performance of the two GP methods we have applied in the experiments.

5.5 Conclusions

In this chapter we have presented a Laplace approximation to make inference in a GP volatility model, where the use of the Ornstein-Uhlenbeck covariance function leads to a plausible model for the process and, at the same time, to a simple (tridiagonal) form for the inverse covariance matrix, which makes the Laplace approximation computationally efficient, requiring $\mathcal{O}(N)$ operations instead the $\mathcal{O}(N^3)$ operations that are needed to compute standard GP models. Moreover, the log-concave likelihood of the volatility model combined with the GP prior leads to a unimodal posterior which guarantees the convergence of the Laplace approximation.

After formalizing this approach, we have checked that the experimental result on a synthetic example shows that the posterior approximation given by LGPVF is accurate when compared to the posterior given by RMHMC. The results obtained for different currency exchange data sets evidence that LGPVF outperforms the commonly used GARCH(1,1) for volatility forecasting tasks. Although the error performance is similar to VHGPR, the computational burden of LGPVF is lower than VHGPR.

Chapter 6

General conclusions and open research lines

In this Thesis, we have developed GP models to perform nonstationary and heteroscedastic regression, using approximate inference techniques, such as EP and the Laplace approximation, and MCMC (with ESS) to make inference on the referred models. On the other hand, we have also proposed a GP method to predict volatility in financial time series, using the Laplace approximation to make inference on the volatility GP model introduced in [Girolami and Calderhead, 2011].

6.1 Contributions and results

- In Chapter 2 we have introduced a DGP model to carry out nonstationary regression including heteroscedastic noise cases. The model is based on the pointwise division of two latent functions, so that a noisy stationary latent function is modulated by another (noise-free) stationary latent function which

describes amplitude nonstationarities affecting both the previous latent function and the noise associated with it. As the likelihood of the DGP model is not Gaussian on both latent functions, the posterior, the evidence, and the predictive distribution cannot be computed analytically. Then, we propose to use EP to perform approximate inference on the model, and ESS to sample from the exact posterior. One of the advantages of the model is that the likelihood is log-concave, which combined with a GP prior leads to a unimodal posterior. This favors EP convergence (although it is not guaranteed), not needing to use any additional tricks, such as damping or skipping, to make the algorithm converge. This contrast with GPPM and the heteroscedastic GP model proposed in [Goldberg et al., 1998], and used later in [Lázaro-Gredilla and Titsias, 2011, Muñoz-González et al., 2011, Quadrianto et al., 2009], among others, where the likelihoods are not log-concave (and then, the posterior distributions are not unimodal).

The MCMC implementation with ESS provides an accurate exact posterior for the DGP, although the computational burden is very high. Yet the EP posterior approximation reduces the computational cost offering a similar performance for regression tasks. The experimental results with unidimensional and multidimensional data sets show the improvements of EP-DGP and MCMC-DGP methods compared with the standard GP, VHGPR, and SVR. The main drawback of EP-DGP is that the computational burden is higher than VHGPR and the standard GP, which limits the application of EP-DGP for big data sets.

- To overcome the limitations of EP-DGP, in Chapter 4 we have proposed to use the Laplace approximation to make inference on the DGP model. Since the posterior of the DGP model is unimodal, the Laplace method converges to a unique maximum. On the other hand, the simplicity of this algorithm reduces the computational burden. Although for some models, such as the standard GP classification, the Laplace method provides poor posterior approximations,

the characteristics of the likelihood in the DGP model allows the proposed L-DGP to obtained similar performance compared to the approximate posterior provided by EP-DGP and the posterior sampled with MCMC-DGP.

The experimental results corroborate the quality of L-DGP, which performs similarly to EP-DGP in most cases, and outperforms VHGPR and the standard GP.

Although EP and the Laplace approximation scale in time as $\mathcal{O}(N^3)$, as it is also the case of the standard GP and VHGPR, the results obtained in the experiments show that the computational burden of L-DGP is lower than EP-DGP and VHGPR.

- In Chapter 5, we have introduced LGPVF, a method to perform inference on the GP volatility model described in [Girolami and Calderhead, 2011] and [Lázaro-Gredilla and Titsias, 2011] using the Laplace approximation. Volatility forecasting is an important task for asset valuation and risk management in financial markets. The commonly used GARCH models to predict volatility are prone to overfitting, as they estimate the parameters of the model by ML. In contrast, GPs are resilient to overfitting.

  The use of the Ornstein-Uhlenbeck covariance function leads to a tridiagonal form for the inverse covariance matrix. This makes the implementation of the Laplace method computationally efficient, requiring $\mathcal{O}(N)$ operations, instead the $\mathcal{O}(N^3)$ achieved by traditional GPs.

  The experimental result on a synthetic example show that the posterior approximation given by LGPVF is accurate when compared to the asymptotically exact posterior given by the MCMC technique proposed in [Girolami and Calderhead, 2011]. The results obtained for different currency exchange data sets show that LGPVF outperforms GARCH(1,1) for volatility forecasting tasks. Although the error performance of LGPVF is similar to VHGPR, the computational burden of LGPVF is much lower.

6.2 Open research lines

Some extensions can be proposed from the DGP model described in Chapters 3-4:

- A nonstandard variational approximation can be proposed to make inference on the DGP model. As in [Lázaro-Gredilla and Titsias, 2011] it is possible to obtain a marginalized lower bound to the evidence integrating out one of the latent functions of the DPG model. This reduces the number of variational parameters and, consequently, lowers the computational burden with respect to the standard variational approach. Although similar performance with respect to EP-DGP and L-DGP is expected, it could be interesting to test if the variational approximation can reduce the computational cost with respect to L-DGP.

- All the proposed algorithms to make inference on the DGP model scale in time as $\mathcal{O}(N^3)$. As in the case of the standard GP, this limits the scope of application of the DGP methods to data sets with a few thousand training samples. Then, it could be useful to apply sparse approximations, as the Sparse Pseudo-input GPs (SPGP) proposed by [Snelson and Ghahramani, 2006], where the covariance is parameterized by the location of M ($< N$) pseudo-input points that can be learned using a gradient based optimization, requiring $\mathcal{O}(M^2N)$ operations to train the model and $\mathcal{O}(M^2)$ operations to make predictions.

- Following the line of sparse Bayesian approximations, Relevance Vector Machines (RVMs) [Tipping, 2001] suffer from impractical, overconfident predictions where the uncertainty tends to be maximal around the training points. To overcome this problem, the algorithm described in [Quiñonero-Candela et al., 2007] aims to decorrelate the prior, which leads to a divisive formulation for the covariance matrix. Since the proposed solution partially solves the aforementioned problem, it could be interesting to develop

an RVM formulation with the DGP model and compare it with the solution proposed in [Quiñonero-Candela et al., 2007].

- Pursuing other types of nonstationarities, it could be interesting to develop a GP model to achieve lengthscale nonstationarities. In [Michalis and Lázaro-Gredilla, 2013] a variational method is proposed to integrate out kernel lengthscale hyperaparameters for GPR, placing a prior density over this hyperparameters and using the variational learning with induced variables described in [Titsias, 2009]. However, it is possible to extend this model and making it more flexible representing the lengthscales as GP random functions, so that these lengthscales are input dependent. Then, this model is suitable for problems in which the smoothness of the underlying latent function varies throughout the input space.

Further research work which we can obtain from the LGPVF method, described in Chapter 5, include:

- The development of other GP models to predict volatility. As described in [Broto and Ruiz, 2004], there are different approaches to model volatility apart from assuming that the noise of the return series is Gaussian. For example, some models describe a different behavior in the return series if the price of the assets rises or falls. Then, given the better predictions of LGPVF compared to GARCH models, it could also be expected a better performance of GPs formulations with respect to the corresponding ML methods used for those other volatility models.

Final note:

The EP-DGP and MCMC-DGP methods that are described in Chapter 3 appear in [Muñoz-González et al., 2014a]. The L-DGP method presented in Chapter 4 is based in [Muñoz-González et al., 2014b]. Finally, the contents of Chapter 5 are based in [Muñoz-González et al., 2014c], where L-GPVF is introduced.

Appendix A

Gaussian identities

A.1 Multivariate Gaussian distribution

The joint probability density of a multivariate (D-dimensional) Gaussian distribution is given by

$$p(\mathbf{x}) = \mathcal{N}(\boldsymbol{x}|\mu, \Sigma) = (2\pi)^{-D/2}|\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \tag{A.1}$$

where $\boldsymbol{\mu}$ is the mean vector of length D and Σ is a symmetric positive-definite covariance matrix of size $D \times D$.

A.2 Marginal and conditional distributions

Let $\mathbf{x}_1, \mathbf{x}_2$ be jointly Gaussian random vectors, so that

$$p(\mathbf{x}_1, \mathbf{x}_2) = \mathcal{N}\left(\begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} \middle| \begin{bmatrix}\boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2\end{bmatrix}, \begin{bmatrix}A & C \\ C^T & B\end{bmatrix}\right) \tag{A.2}$$

The marginal distribution for $\mathbf{x}_1$ is given by

$$p(\mathbf{x}_1) = \mathcal{N}(\mathbf{x}_1|\boldsymbol{\mu}_1, A) \tag{A.3}$$

and the conditional distribution of $\mathbf{x}_1$ given $\mathbf{x}_2$ is

$$p(\mathbf{x}_1|\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_1|\boldsymbol{\mu}_1 + CB^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2), A - CB^{-1}C^T) \tag{A.4}$$

A.3 Product of two Gaussian distributions

The product of two Gaussian distributions, $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_1, \Sigma_1)$ and $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_2, \Sigma_2)$ results in an un-normalized Gaussian distribution given by

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_1, \Sigma_1)\, \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_2, \Sigma_2) = Z^{-1}\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \Sigma) \tag{A.5}$$

where $\Sigma = (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1}$, $\boldsymbol{\mu} = \Sigma(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2)$, and the normalizing constant is like a Gaussian (on $\boldsymbol{\mu}_1$ or $\boldsymbol{\mu}_2$)

$$Z^{-1} = (2\pi)^{-D/2}|\Sigma_1 + \Sigma_2|^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T(\Sigma_1 + \Sigma_2)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)\right) \tag{A.6}$$

Appendix B

Matrix algebra

B.1 Matrix inversion lemma

The Woodbury, Sherman and Morrison formula, better known as the matrix inversion lemma, states that

$$(A + BCD^T)^{-1} = A^{-1} - A^{-1}B(C^{-1} + D^T A^{-1} B)^{-1} D^T A^{-1} \tag{B.1}$$

provided that the relevant inverses all exist, where A is $n \times n$, C is $m \times m$, and B and D are both $n \times m$.

If A^{-1} is known and $m < n$, a considerable speedup can be achieved applying the right hand side of (B.1). For example, if A is diagonal and B, C, and D are full matrices, direct evaluation of the left hand side of (B.1) is $\mathcal{O}(n^3)$, whereas the right hand side can be evaluated only in $\mathcal{O}(m^2 n)$.

B.2 Matrix determinant lemma

According to the above, the matrix determinant lemma provides the following equation

$$|A + BCD^T| = |A| \cdot |C| \cdot |C^{-1} + D^T A^{-1} B| \tag{B.2}$$

B.3 Matrix derivatives

The derivative of the elements of the inverse matrix B^{-1} with respect to a scalar parameter α is given by

$$\frac{\partial B^{-1}}{\partial \alpha} = -B^{-1} \frac{\partial B}{\partial \alpha} B^{-1} \tag{B.3}$$

where $\partial B / \partial \alpha$ is a matrix of elementwise derivatives.

If B is a positive definite symmetric matrix, the derivative of the log-determinant is calculated as

$$\frac{\partial \log |B|}{\partial \alpha} = \mathrm{tr}\left(B^{-1} \frac{\partial B}{\partial \alpha} \right) \tag{B.4}$$

For more details on matrix differential calculus see [Lütkepohl, 1996, Harville, 1997].

B.4 Cholesky factorization

The Cholesky factorization or Cholesky decomposition of a symmetric, positive definite matrix B consists on the factorization of the given matrix into a product of a lower triangular matrix L and its transpose

$$B = LL^T \tag{B.5}$$

where L is the Cholesky factor of B. The computation of L scales as $\mathcal{O}(N^3/6)$, and it is numerically very stable. It is a special case of LU matrix factorization when the

matrix to be factorized is symmetric and positive definite.

The Cholesky factorization is very useful for solving linear systems, $B\mathbf{x} = \mathbf{c}$, when the coefficient matrix B is symmetric and positive definite. In that case the solution can be written as

$$\mathbf{x} = L^T \backslash (L \backslash \mathbf{c}) \tag{B.6}$$

where $D\backslash d$ is the solution of the linear system $D\mathbf{x} = d$. The, the solution of the two linear systems can be computed efficiently in $\mathcal{O}(N^2/2)$ time using forward and backward substitution, performing faster than directly solving $B\mathbf{x} = \mathbf{c}$.

The Cholesky decomposition is also useful to calculate the determinant of a symmetric positive definite matrix, which can be expressed as

$$|B| = \prod_{n=1}^{N} L_{nn}^2 \tag{B.7}$$

or equivalently

$$\log |B| = 2 \sum_{n=1}^{N} \log L_{nn} \tag{B.8}$$

which, in many cases, is a more precise way to compute $|B|$ than computing (B.7).

Appendix C

EP-DGP calculations

C.1 Details of EP-DGP moments calculations

To simplify the calculations of the EP-DGP moments, we can express the likelihood given in (3.2) as:

$$p(y_n|f_n, g_n) = g_n^+ \mathcal{N}(f_n|y_n g_n^+, c) \tag{C.1}$$

C.1.1 Zeroth order moment

The zeroth order moment in (3.15) can be calculated as

$$\begin{aligned}
\hat{Z}_n &= \iint \mathcal{N}(f_n|\mu_{f_{\setminus n}}, \sigma^2_{f_{\setminus n}}) \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma^2_{g_{\setminus n}})\ g_n^+ \mathcal{N}(f_n|y_n g_n^+, c)\ df_n\ dg_n \\
&= \int_0^\infty \frac{g_n}{|y_n|} \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma^2_{g_{\setminus n}})\ \mathcal{N}\left(g_n|\frac{\mu_{f_{\setminus n}}}{y_n}, \frac{c+\sigma^2_{f_{\setminus n}}}{y_n^2}\right) dg_n \\
&= \breve{Z} \int_0^\infty g_n \mathcal{N}(g_n|\breve{\mu}_g, \breve{\sigma}^2_g) dg_n = \breve{Z}\ \breve{\mu}_{g_t}\ \Phi\left(-\frac{\breve{\mu}_g}{\breve{\sigma}_g}\right)
\end{aligned} \tag{C.2}$$

The mean of the Gaussian $\mathcal{N}(g_n|\breve{\mu}_g, \breve{\sigma}_g^2)$ truncated to the positive values of g_n can be expressed as

$$\breve{\mu}_{g_t} = \breve{\mu}_g + \breve{\sigma}_g \lambda \left(\frac{\breve{\mu}_g}{\breve{\sigma}_g} \right) \tag{C.3}$$

where

$$\lambda(x) = \frac{\varphi(x)}{\Phi(-x)} \tag{C.4}$$

$$\varphi(x) = \frac{1}{2\pi} e^{-\frac{x^2}{2}} \tag{C.5}$$

C.1.2 First order moments

The calculation of the first order moment of g_n can be written as

$$\begin{aligned} \hat{\mu}_{g_n} &= \frac{1}{\hat{Z}_n} \iint g_n \mathcal{N}(f_n|\mu_{f_{\setminus n}}, \sigma^2_{f_{\setminus n}})\, \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma^2_{g_{\setminus n}}) g_n^+ \mathcal{N}(f_n|y_n g_n^+, c)\, df_n\, dg_n \\ &= \frac{1}{\hat{Z}_n} \int_0^\infty \frac{g_n^2}{|y_n|} \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma^2_{g_{\setminus n}})\, \mathcal{N}\left(g_n | \frac{\mu_{f_{\setminus n}}}{y_n}, \frac{c + \sigma^2_{f_{\setminus n}}}{y_n^2} \right) dg_n \\ &= \frac{\breve{Z}}{\hat{Z}_n} \int_0^\infty g_n^2 \mathcal{N}(g_n|\breve{\mu}_g, \breve{\sigma}_g^2) dg_n = \frac{\breve{Z}}{\hat{Z}_n}\, m_{2_t}\, \Phi\left(-\frac{\breve{\mu}_g}{\breve{\sigma}_g} \right) \end{aligned} \tag{C.6}$$

where the non-centered second order moment of the Gaussian $\mathcal{N}(g_n|\breve{\mu}_g, \breve{\sigma}_g^2)$ truncated to the positive values of g_n is calculated as

$$m_{2_t} = \breve{\mu}_g^2 + 2\breve{\mu}_g \breve{\sigma}_g \lambda \left(\frac{\breve{\mu}_g}{\breve{\sigma}_g} \right) + \breve{\sigma}_g^2 \left(1 - \frac{\breve{\mu}_g}{\breve{\sigma}_g} \lambda \left(\frac{\breve{\mu}_g}{\breve{\sigma}_g} \right) \right) \tag{C.7}$$

For the first order moment of f_n we have

$$\begin{aligned} \hat{\mu}_{f_n} &= \frac{1}{\hat{Z}_n} \iint f_n \mathcal{N}(f_n|\mu_{f_{\setminus n}}, \sigma^2_{f_{\setminus n}})\, \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma^2_{g_{\setminus n}}) g_n^+ \mathcal{N}(f_n|y_n g_n^+, c)\, df_n\, dg_n \\ &= \frac{1}{\hat{Z}_n} \int_0^\infty \frac{g_n}{|y_n|} \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma^2_{g_{\setminus n}}) \breve{\mu}_f\, \mathcal{N}\left(g_n | \frac{\mu_{f_{\setminus n}}}{y_n}, \frac{c + \sigma^2_{f_{\setminus n}}}{y_n^2} \right) dg_n \end{aligned} \tag{C.8}$$

where

$$\breve{\mu}_f = \breve{\sigma}_f^2 \left(\frac{|y_n| g_n^+}{c} + \frac{\mu_{f_{\setminus n}}}{\sigma_{f_{\setminus n}}^2} \right) \qquad \text{(C.9)}$$
$$\breve{\sigma}_f^2 = (c^{-1} + \sigma_{f_{\setminus n}}^{-2})^{-1}$$

Note that $\breve{\mu}_f$ depends on g_n. Then, the solution of integral (C.8) involves the calculation of the first and second order moments of the truncated Gaussian $\mathcal{N}(g_n|\breve{\mu}_g, \breve{\sigma}_g^2)$, which results in the solution given in (3.19).

C.1.3 Second order moments

The non-centered second order moment of g_n can be calculated as

$$\begin{aligned} \hat{m}_{2_{g_n}} &= \frac{1}{\hat{Z}_n} \iint g_n^2 \mathcal{N}(f_n|\mu_{f_{\setminus n}}, \sigma_{f_{\setminus n}}^2)\, \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma_{g_{\setminus n}}^2) g_n^+ \mathcal{N}(f_n|y_n g_n^+, c)\, df_n\, dg_n \\ &= \frac{1}{\hat{Z}_n} \int_0^\infty \frac{g_n^3}{|y_n|} \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma_{g_{\setminus n}}^2)\, \mathcal{N}\left(g_n | \frac{\mu_{f_{\setminus n}}}{y_n}, \frac{c + \sigma_{f_{\setminus n}}^2}{y_n^2} \right) dg_n \\ &= \frac{\breve{Z}}{\hat{Z}_n} \int_0^\infty g_n^3 \mathcal{N}(g_n|\breve{\mu}_g, \breve{\sigma}_g^2) dg_n = \frac{\breve{Z}}{\hat{Z}_n}\, m_{3_t}\, \Phi\left(-\frac{\breve{\mu}_g}{\breve{\sigma}_g} \right) \end{aligned} \qquad \text{(C.10)}$$

where the non-centered third order moment of the Gaussian $\mathcal{N}(g_n|\breve{\mu}_g, \breve{\sigma}_g^2)$ truncated to the positive values of g_n is expressed as

$$m_{3_t} = \breve{\mu}_g^3 + 3\breve{\mu}_g^2 \breve{\sigma}_g \lambda\left(\frac{\breve{\mu}_g}{\breve{\sigma}_g} \right) + 3\breve{\mu}_g \breve{\sigma}_g^2 \left(1 - \frac{\breve{\mu}_g}{\breve{\sigma}_g} \lambda\left(\frac{\breve{\mu}_g}{\breve{\sigma}_g} \right) \right) + \breve{\sigma}_g^3 \lambda\left(\frac{\breve{\mu}_g}{\breve{\sigma}_g} \right) \left(2 + \frac{\breve{\mu}_g^2}{\breve{\sigma}_g^2} \right) \qquad \text{(C.11)}$$

Finally, the expression of the second order moment of f_n is given by

$$\begin{aligned} \hat{m}_{2_{f_n}} &= \frac{1}{\hat{Z}_n} \iint f_n^2 \mathcal{N}(f_n|\mu_{f_{\setminus n}}, \sigma_{f_{\setminus n}}^2)\, \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma_{g_{\setminus n}}^2) g_n^+ \mathcal{N}(f_n|y_n g_n^+, c)\, df_n\, dg_n \\ &= \frac{1}{\hat{Z}_n} \int_0^\infty \frac{g_n}{|y_n|} \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma_{g_{\setminus n}}^2)(\breve{\mu}_f^2 + \breve{\sigma}_f^2)\, \mathcal{N}\left(g_n | \frac{\mu_{f_{\setminus n}}}{y_n}, \frac{c + \sigma_{f_{\setminus n}}^2}{y_n^2} \right) dg_n \\ &= \frac{1}{\hat{Z}_n} \int_0^\infty \mathcal{N}(g_n|\mu_{g_{\setminus n}}, \sigma_{g_{\setminus n}}^2) \mathcal{N}\left(g_n | \frac{\mu_{f_{\setminus n}}}{y_n}, \frac{c + \sigma_{f_{\setminus n}}^2}{y_n^2} \right) \times \\ &\quad \left(\frac{g_n \breve{\sigma}_f^2}{|y_n|} + \frac{g_n^3 y_n \breve{\sigma}_f^4}{c^2} + \frac{g_n \mu_{f_{\setminus n}}^2 \breve{\sigma}_f^4}{|y_n| \sigma_{f_{\setminus n}}^4} + 2\frac{g_n^2 \breve{\sigma}_f^4 \mu_{f_{\setminus n}}}{c \sigma_{f_{\setminus n}}^2} \right) dg_n \end{aligned} \qquad \text{(C.12)}$$

As in the case of the calculation of $\hat{\mu}_{f_n}$, the term $\breve{\mu}_f$ depends on g_n, so that some calculations with moments of the truncated Gaussian to the positive values are needed to solve the integral, leading to the solution in (3.21) after straightforward simplifications.

C.2 Derivatives of the log-evidence with respect to the hyperparameters

Now we detail the calculations of the partial derivatives of the log-evidence with respect to the hyperparameters $\boldsymbol{\theta}_f$,$\boldsymbol{\theta}_g$, and μ_0. These derivatives are necessary to search the values of the hyperparameters that maximize the log-evidence using ML-II. The expressions obtained are similiar to those described in [Rasmussen and Williams, 2006] for GP classification using EP. The derivatives of $\boldsymbol{\theta}_f$, $\boldsymbol{\theta}_g$, include explicit derivatives corresponding to the terms of the log-evidence which depend on K_f and K_g, respectively, and implicit derivatives in the rest of the terms, due to the site EP estimation for the site functions. Fortunately, implicit derivatives are exactly zero, as proven in [Seeger, 2005], so we only need to calculate the explicit terms.

The partial derivatives with respect to each of the hyperparameters in $\boldsymbol{\theta}_f$ are given by:

$$\begin{aligned}\frac{\partial}{\partial\theta_{f_j}}\log q(\mathbf{y}) &= \frac{1}{2}\mathbf{y}^T(K_f+\tilde{\Sigma}_f)^{-1}\frac{\partial K_f}{\partial\theta_{f_j}}(K_f+\tilde{\Sigma}_f)^{-1}\mathbf{y} - \frac{1}{2}\mathrm{tr}\left((K_f+\tilde{\Sigma}_f)^{-1}\frac{\partial K_f}{\partial\theta_{f_j}}\right)\\ &= \frac{1}{2}\mathrm{tr}\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^T-(K_f+\tilde{\Sigma}_f)^{-1})\frac{\partial K_f}{\partial\theta_{f_j}}\right)\end{aligned} \tag{C.13}$$

where $\boldsymbol{\alpha} = (K_f+\tilde{\Sigma}_f)^{-1}\mathbf{y}$. The calculation of these derivatives requires $\mathcal{O}(N^3)$ operations to invert the symmetric matrix $(K_g+\tilde{\Sigma}_g)$, and $\mathcal{O}(N^2)$ operations to

compute each partial derivative with respect to $\boldsymbol{\theta}_f$ [Rasmussen and Williams, 2006].

For the set of hyperparameters $\boldsymbol{\theta}_g$, we have a similar expression:

$$\begin{aligned}\frac{\partial \log q(\mathbf{y})}{\partial \theta_{g_j}} &= \frac{\partial}{\partial \theta_{g_j}} - \frac{1}{2}\log|K_g + \tilde{\Sigma}_g| - \frac{1}{2}(\boldsymbol{\mu}_0 - \tilde{\boldsymbol{\mu}}_g)^T(K_g + \tilde{\Sigma}_g)^{-1}(\boldsymbol{\mu}_0 - \tilde{\boldsymbol{\mu}}_g) \\ &= \frac{1}{2}(\boldsymbol{\mu}_0 - \tilde{\boldsymbol{\mu}}_g)^T(K_g + \tilde{\Sigma}_g)^{-1}\frac{\partial K_g}{\partial \theta_{g_j}}(K_g + \tilde{\Sigma}_g)^{-1}\ (\boldsymbol{\mu}_0 - \tilde{\boldsymbol{\mu}}_g) - \\ &\quad \frac{1}{2}\mathrm{tr}\left((K_g + \tilde{\Sigma}_g)^{-1}\frac{\partial K_g}{\partial \theta_{g_j}}\right) \\ &= \frac{1}{2}\mathrm{tr}\left((\boldsymbol{\beta}\boldsymbol{\beta}^T - (K_g + \tilde{\Sigma}_g)^{-1})\frac{\partial K_g}{\partial \theta_{g_j}}\right)\end{aligned} \tag{C.14}$$

where $\boldsymbol{\beta} = (K_g + \tilde{\Sigma}_g)^{-1}(\boldsymbol{\mu}_0 - \tilde{\boldsymbol{\mu}}_g)$. The computational complexity of these calculations is also $\mathcal{O}(N^3)$ to invert $(K_g + \tilde{\Sigma}_g)$ and $\mathcal{O}(N^2)$ for each partial derivative. Finally, the derivative with respect to the hyperparameter μ_0 can be written as:

$$\begin{aligned}\frac{\partial}{\partial \mu_0}\log q(\mathbf{y}) &= \frac{\partial}{\partial \mu_0}\left(-\frac{1}{2}(\boldsymbol{\mu}_0 - \tilde{\boldsymbol{\mu}}_g)^T(\mathbf{K}_g + \tilde{\boldsymbol{\Sigma}}_g)^{-1}(\boldsymbol{\mu}_0 - \tilde{\boldsymbol{\mu}}_g)\right) \\ &= -(\boldsymbol{\mu}_0 - \tilde{\boldsymbol{\mu}}_g)^T(\mathbf{K}_g + \tilde{\boldsymbol{\Sigma}}_g)^{-1}\mathbf{1}_N = -\boldsymbol{\beta}^T\mathbf{1}_N\end{aligned} \tag{C.15}$$

This derivative does not imply additional relevant computational cost, as $\boldsymbol{\beta}$ is also needed for the derivatives on $\boldsymbol{\theta}_g$.

C.3 DGP predictive distribution calculation

The predictive distribution for a new test sample y_*, given in (3.29), can be calculated as follows

$$\begin{aligned}q(y_*) &= \iint \mathcal{N}(f_*|\mu_{f_*}, \sigma^2_{f_*})\mathcal{N}(g_*|\mu_{g_*}, \sigma^2_{g_*})\ g_*^+\mathcal{N}(f_*|y_* g_*^+, c)\ df_*\ dg_* \\ &= \int_0^\infty \frac{g_*}{|y_*|}\mathcal{N}(g_*|\mu_{g_*}, \sigma^2_{g_*})\ \mathcal{N}\left(g_*|\frac{\mu_{f_*}}{y_*}, \frac{c + \sigma^2_{f_*}}{y_*^2}\right) dg_* \\ &= Z_*(y_*)\int_0^\infty g_*\mathcal{N}(g_*|\breve{\mu}_{g_*}, \breve{\sigma}^2_{g_*})dg_* \\ &= Z_*(y_*)\ \breve{\mu}_{g_{*_t}}\ \Phi\left(-\frac{\breve{\mu}_{g_*}}{\breve{\sigma}_{g_*}}\right)\end{aligned} \tag{C.16}$$

where $\breve{\mu}_{g_{*_t}}$ is the mean of the Gaussian $\mathcal{N}(g_*|\breve{\mu}_{g_*}, \breve{\sigma}^2_{g_*})$ truncated to the positive values of g_*, given by

$$\breve{\mu}_{g_{*_t}} = \breve{\mu}_{g_*} + \breve{\sigma}_{g_*} \lambda \left(\frac{\breve{\mu}_{g_*}}{\breve{\sigma}_{g_*}} \right) \tag{C.17}$$

Appendix D

Derivatives of the log-evidence for L-DGP

In this appendix, we detail the calculations of the partial derivatives of the approximate log-evidence for L-DGP, given in (4.13), with respect to the hyperparameters of the covariance functions, $\boldsymbol{\theta}_f$ and $\boldsymbol{\theta}_g$, and the noise offset μ_0 used in the likelihood. These derivatives are necessary for the ML-II implementation to search the hyperparameters that maximize the approximate log-evidence.

To calculate the derivatives we have to take into account not only the explicit terms, i.e. those referred to K in the case of $\boldsymbol{\theta}_f$ and $\boldsymbol{\theta}_g$ or to the likelihood in the case of μ_0, but also the implicit derivatives, since when the hyperparameters change, the optimum of the posterior $\hat{\boldsymbol{\phi}}$ also changes.

The derivative of the approximate log-evidence with respect to each hyperparameter in $\boldsymbol{\theta}_f$ can be expressed as:

$$\frac{\partial \log q(\mathbf{y})}{\partial \theta_{f_j}} = \left.\frac{\partial \log q(\mathbf{y})}{\partial \theta_{f_j}}\right|_{\text{explicit}} + \sum_{n=0}^{N} \frac{\partial \log q(\mathbf{y})}{\partial \hat{f}_n} \frac{\partial \hat{f}_n}{\theta_{f_j}} + \sum_{n=0}^{N} \frac{\partial \log q(\mathbf{y})}{\partial \hat{g}_n} \frac{\partial \hat{g}_n}{\theta_{f_j}} \tag{D.1}$$

The explicit derivative is given by

$$\left.\frac{\partial \log q(\mathbf{y})}{\partial \theta_{f_j}}\right|_{\text{explicit}} = \frac{1}{2}\hat{\phi}^T K^{-1}\frac{\partial K}{\partial \theta_{f_j}}K^{-1}\hat{\phi} - \frac{1}{2}\text{tr}\left[(W^{-1}+K)^{-1}\frac{\partial K}{\partial \theta_{f_j}}\right] \tag{D.2}$$

To calculate the implicit derivatives we have:

$$\frac{\partial \log q(\mathbf{y})}{\partial \hat{f}_n} = -\frac{1}{2}\left[(K^{-1}+W)^{-1}\right]_{nn}\frac{\partial^3 \log p(y_n|\hat{\phi}_n)}{\partial f_n^3} \tag{D.3}$$

which is equal to zero as the third derivative of the local likelihood with respect to f_n is zero. For g_n, the expression is analogous to (D.3), but in this case, the third derivative of the local likelihood with respect to g_n is given by

$$\frac{\partial^3 \log p(y_n|\hat{\phi}_n)}{\partial g_n^3} = \begin{cases} \dfrac{2}{g_n'^3}, & \text{if } g_n' > 0 \\ 0, & \text{if } g_n' \leq 0 \end{cases} \tag{D.4}$$

with $g_n' = g_n + \mu_0$. Hence, in contrast to the case of f_n, the implicit derivatives with respect to g_n do not vanish for positive g_n'.

To compute $\partial \hat{f}_n/\theta_{f_j}$ we first calculate $\partial\hat{\phi}/\partial\theta_{f_j}$ and then select the corresponding terms for each g_n, so that

$$\frac{\partial \hat{\phi}}{\partial \theta_{f_j}} = (I + KW)^{-1}\frac{\partial K}{\partial \theta_{f_j}}\nabla \log p(\mathbf{y}|\hat{\phi}) \tag{D.5}$$

The calculations of the derivatives with respect to each hyperparameter in $\boldsymbol{\theta}_g$ are analogous to those for $\boldsymbol{\theta}_f$.

The derivative with respect to the mean offset μ_0 for latent function g can be expressed

$$\frac{\partial \log q(\mathbf{y})}{\partial \mu_0} = \left.\frac{\partial \log q(\mathbf{y})}{\partial \mu_0}\right|_{\text{explicit}} + \sum_{n=0}^{N}\frac{\partial \log q(\mathbf{y})}{\partial \hat{g}_n}\frac{\partial \hat{g}_n}{\partial \mu_0} \tag{D.6}$$

where the explicit derivative can be calculated as

$$\left.\frac{\partial \log q(\mathbf{y})}{\partial \mu_0}\right|_{\text{explicit}} = \sum_{n=0}^{N}\frac{\partial \log p(y_n|\hat{\phi}_n)}{\partial \mu_0} - \frac{1}{2}\frac{\partial \log |B|}{\partial \mu_0} \tag{D.7}$$

with

$$\frac{\partial \log p(y_n|\hat{\phi}_n)}{\partial \mu_0} = \begin{cases} \frac{1}{g'_n} - \frac{y_n^2 g'_n - y_n f_n}{c}, & g'_n > 0 \\ 0, & g'_n \leq 0 \end{cases} \tag{D.8}$$

and

$$-\frac{1}{2}\frac{\partial \log |B|}{\partial \mu_0} = -\frac{1}{2}\text{tr}\left(B^{-1}K\frac{\partial W}{\partial \mu_0}\right) \tag{D.9}$$

where $\partial W/\partial \mu_0$ is a $2N$ diagonal matrix with 0 in the first N elements of the diagonal, and

$$\left[\frac{\partial W}{\partial \mu_0}\right]_{nn} = \begin{cases} \frac{2}{g'^3_{n'}}, & \text{if } g'_{n'} > 0 \\ 0, & \text{if } g'_{n'} \leq 0 \end{cases} \tag{D.10}$$

for $N < n \leq 2N$ with $n' = n - N$.

Finally, to calculate the implicit derivative, along with the expression for $\partial \log q(\mathbf{y})/\partial \hat{g}_n$ calculated in (D.3) and (D.4), we have

$$\frac{\partial \hat{g}_n}{\partial \mu_0} = (I + KW)^{-1}K\frac{\partial \nabla \log p(\mathbf{y}|\hat{\phi})}{\partial \mu_0} \tag{D.11}$$

where

$$\frac{\partial \nabla \log p(\mathbf{y}|\hat{\phi})}{\partial \mu_0} = -\text{diag}(W) \tag{D.12}$$

Appendix E

Derivatives of the log-evidence for LGPVF

To select the hyperparameters of LGPVF, we apply a Maximum Likelihood of level II (ML-II) procedure, where a (possibly local) maximum of the log-evidence with respect to the hyperparameters is sought.

The approximate log-evidence for LGPVF, as also shown in Section 5.5, is given by

$$\log q(\mathbf{y}) = -\frac{1}{2}\hat{\mathbf{g}}^T K_g^{-1}\hat{\mathbf{g}} + \log p(\mathbf{y}|\hat{\mathbf{g}}) - \frac{1}{2}\log(|B|) \tag{E.1}$$

where $B = I + WK_gW$, $\hat{\mathbf{g}}$ is the maximum of the posterior found by the Newton method used to solve the self-consistent equation given in (5.13), and W is the negative Hessian of the log-likelihood, i.e., $W = -\nabla\nabla \log p(\mathbf{y}|\mathbf{g})$.

To calculate the partial derivatives of the log-evidence (E.1) with respect to the hyperparameters of the colvariance function, $\boldsymbol{\theta}$, it is important to note that not only covariance matrix K_g depends on these hyperparameters but $\hat{\mathbf{g}}$ and W are also implicitly functions of $\boldsymbol{\theta}$, since if $\boldsymbol{\theta}$ changes, then $\hat{\mathbf{g}}$ also changes. Thus, the partial

derivative with respect to each hyperparameter in the covariance function can be written as

$$\frac{\partial \log q(\mathbf{y})}{\partial \theta_j} = \frac{\partial \log q(\mathbf{y})}{\partial \theta_j}\bigg|_{\text{explicit}} + \sum_{n=0}^{N} \frac{\partial \log q(\mathbf{y})}{\partial \hat{g}_n} \frac{\partial \hat{g}_n}{\theta_j} \tag{E.2}$$

Then, the explicit derivative can be calculated using the chain rule, leading to

$$\frac{\partial \log q(\mathbf{y})}{\partial \theta_j}\bigg|_{\text{explicit}} = \frac{1}{2}\hat{\mathbf{g}}^T K_g^{-1} \frac{\partial K_g}{\partial \theta_j} K_g^{-1} \hat{\mathbf{g}} - \frac{1}{2}\text{tr}\left[(W^{-1} + K_g)^{-1} \frac{\partial K_g}{\partial \theta_j}\right] \tag{E.3}$$

To evaluate the implicit derivatives from (E.2), we use the fact that $\nabla\Psi(\mathbf{g}) = \mathbf{0}$ at $\mathbf{g} = \hat{\mathbf{g}}$, where $\Psi(\mathbf{g})$ is defined in equation (5.12). This makes that the implicit derivatives of the first two terms in (E.1) vanish. So that,

$$\begin{aligned} \frac{\partial \log q(\mathbf{y})}{\partial \hat{g}_n} &= -\frac{1}{2}\frac{\partial \log |B|}{\partial \hat{g}_n} = -\frac{1}{2}\text{tr}\left(B^{-1} K_g \frac{\partial W}{\partial \hat{g}_n}\right) \\ &= -\frac{1}{2}\left[(K_g^{-1} + W)^{-1}\right]_{nn} \frac{\partial^3 \log p(\mathbf{y}|\hat{\mathbf{g}})}{\partial g_n^3} \end{aligned} \tag{E.4}$$

where

$$\frac{\partial^3 \log p(\mathbf{y}|\hat{\mathbf{g}})}{\partial g_n^3} = \frac{y_n^2}{2e^{(\hat{g}_n + \mu_0)}} \tag{E.5}$$

We can calculate the remaining term $\partial\hat{\mathbf{g}}/\partial\theta_j$ differentiating the self-consistent equation (5.13) $\hat{\mathbf{g}} = K_g(\nabla \log p(\mathbf{y}|\hat{\mathbf{g}}))$ with respect to θ_j:

$$\begin{aligned} \frac{\partial \hat{\mathbf{g}}}{\theta_j} &= \frac{\partial K_g}{\partial \theta_j} \nabla \log p(\mathbf{y}|\hat{\mathbf{g}}) + K_g \frac{\partial \nabla \log p(\mathbf{y}|\hat{\mathbf{g}})}{\partial \hat{\mathbf{g}}} \frac{\partial \hat{\mathbf{g}}}{\partial \theta_j} \\ &= (I + K_g W)^{-1} \frac{\partial K_g}{\partial \theta_j} \nabla \log p(\mathbf{y}|\hat{\mathbf{g}}) \end{aligned} \tag{E.6}$$

Finally, the derivative of the log-evidence with respect to μ_0 (which is a hyper-parameter of the likelihood) is

$$\frac{\partial \log q(\mathbf{y})}{\partial \mu_0} = \frac{\partial \log q(\mathbf{y})}{\partial \mu_0}\bigg|_{\text{explicit}} + \sum_{n=0}^{N} \frac{\partial \log q(\mathbf{y})}{\partial \hat{g}_n} \frac{\partial \hat{g}_n}{\partial \mu_0} \tag{E.7}$$

where the explicit term can be calculated as

$$\left.\frac{\partial \log q(\mathbf{y})}{\partial \mu_0}\right|_{\text{explicit}} = \sum_{n=0}^{N} \frac{\partial \log p(y_n|\hat{\mathbf{g}}_n)}{\partial \mu_0} - \frac{1}{2}\frac{\partial \log |B|}{\partial \mu_0} \tag{E.8}$$

with

$$\frac{\partial \log p(y_n|\hat{g}_n)}{\partial \mu_0} = \frac{y_n^2}{2e^{(\hat{g}_n+\mu_0)}} \tag{E.9}$$

and

$$-\frac{1}{2}\frac{\partial \log |B|}{\partial \mu_0} = -\frac{1}{2}\text{tr}\left(B^{-1}K_g\frac{\partial W}{\partial \mu_0}\right) \tag{E.10}$$

where $\partial W/\partial \mu_0$ is an N diagonal matrix with

$$\left[\frac{\partial W}{\partial \mu_0}\right]_{nn} = -\frac{y_n^2}{2e^{(\hat{g}_n+\mu_0)}} \tag{E.11}$$

To calculate the implicit derivative, we use the expression for $\partial \log q(\mathbf{y})/\partial \hat{g}_n$ calculated in (E.4), and for $\partial \hat{g}_n/\partial \mu_0$, we use the same procedure than in equation (E.6), obtaining

$$\frac{\partial \hat{g}_n}{\partial \mu_0} = (I + K_g W)^{-1}\frac{\partial K_g}{\partial \theta_j}\,\nabla \log p(\mathbf{y}|\hat{\mathbf{g}}) \tag{E.12}$$

where

$$\frac{\partial \nabla \log p(\mathbf{y}|\hat{\mathbf{g}})}{\partial \mu_0} = -\text{diag}(W) \tag{E.13}$$

Appendix F

Wilcoxon rank-sum test

Given N_A and N_B samples from populations A and B, respectively, the Wilcoxon rank-sum test is based on the ranking of the $N_A + N_B$ samples. Each observation has a rank: The smallest sample has rank 1 and the largest sample has rank $N_A + N_B$. Then, the Wilcoxon rank-sum test statistic is the sum of the ranks of the samples from one of the populations.

Assuming that the null hypotheses, H_0, corresponds to the case where the the two populations are equal, the p-value is defined as the probability of H_0 to be true. Defining w_A as the observed rank-sum for the samples in A and W_A as the random variable that models the rank-sum for the samples in A, the p-value is given by

$$p = 2\Pr(W_A \geq w_A) \tag{F.1}$$

if the mean of the samples in A is greater to the mean of samples in B, or

$$p = 2\Pr(W_A \leq w_A) \tag{F.2}$$

in other case.

For samples sizes bigger than 10 on both populations, we can approximate the distribution of W_A as a Gaussian, $\mathcal{N}(\mu_A, \sigma_A^2)$, where

$$\mu_A = \frac{N_A(N_A + N_B + 1)}{2} \tag{F.3}$$

$$\sigma_A = \sqrt{\frac{N_A N_B(N_A + N_B + 1)}{12}} \tag{F.4}$$

So that, the p-value in (F.1) or (F.2) can be computed by means of the cumulative Gaussian distribution.

The p-value is interpreted as a measure of the statistical evidence that the two populations support the null hypotheses: There is no statistical difference between the two populations. The null hypotheses is rejected when the p-value turns out to be less that a predetermined significance level, typically 0.05.

For a more complete description of these statistical tests see [Hollander et al., 1999].

Bibliography

[Adams and Stegle, 2008] Adams, R. P. and Stegle, O. (2008). Gaussian process product models for nonparametric nonstationarity. In McCallum, A. and Roweis, S., editors, *Proceedings of the 25th International Conference on Machine Learning*, pages 1–8, Helsinki. Omnipress.

[Aizerman et al., 1964] Aizerman, M. A., Braverman, E. M., and Rozonoehr, L. I. (1964). The probability problem of pattern recognition learning and the method of potential functions. *Automation and Remote Control*, 25:1175–1190.

[Bache and Lichman, 2014] Bache, K. and Lichman, M. (2014). UCI Machine Learning Repository, http://archive.ics.uci.edu/ml. School of Information and Computer Sciences, University of California at Irvine, CA.

[Bengio et al., 2007] Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2007). Greedy layer-wise training of deep networks. In Schölkopf, B., Platt, J. C., and Hoffman, T., editors, *Advances in Neural Information Processing Systems*, volume 19, pages 153–160. MIT Press.

[Bollerslev, 1986] Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31:307–327.

[Boser et al., 1992] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Haussler, D., editor, *Proceedings of the*

5th Annual Workshop on Computational Learning Theory, pages 144–152, New York, NY.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine Learning*, 45:5–32.

[Brooks et al., 2001] Brooks, C., Burke, S. P., and Persand, G. (2001). Benchmarks and the accuracy of GARCH model estimation. *International Journal of Forecasting*, 17:45–56.

[Broto and Ruiz, 2004] Broto, C. and Ruiz, E. (2004). Estimation methods for stochastic volatility models: A survey. *Journal of Economic Surveys*, 18:613–649.

[Brownlees et al., 2011] Brownlees, C., Engle, R., and Kelly, B. (2011). A practical guide to volatility forecasting through calm and storm. *Journal of Risk*, 14:1–20.

[Cawley et al., 2006] Cawley, G., Talbot, N., and Chapelle, O. (2006). Estimating predictive variances with kernel ridge regression. In Quiñonero-Candela, J., Dagan, I., Magnini, B., and D'Alché-Buc, F., editors, *Machine Learning Challenges: Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment*, volume 3944, Lecture Notes in Computer Science, pages 56–77, Berlin-Heidelberg. Springer.

[Cortez et al., 2009] Cortez, P., Cerdeira, A., Almeida, F., Matos, T., and Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47:547–553.

[Cybenko, 1989] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics Control, Signals, and Systems*, 2:303–314.

[Drucker et al., 1997] Drucker, H., Burges, C. J. C., Kaufman, L., Smola, A., and Vapnik, V. (1997). Support vector regression machines. In Mozer, M. C., Jor-

dan, M. I., and Petsche, T., editors, *Advances in Neural Information Processing Systems*, volume 9, pages 155–161. Morgan Kaufmann, San Mateo, CA.

[Efron et al., 2004] Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, 32:407–499.

[Engle, 1983] Engle, R. F. (1983). Estimates of the variance of US inflation based upon the ARCH model. *Journal of Money, Credit, and Banking*, 15:286–301.

[Figueiras Vidal, 2013] Figueiras Vidal, A. R. (2013). *De Máquinas y Humanos. El Arte de la Toma de Decisiones.* Real Academia de Ingeniería, Madrid.

[Freund, 1995] Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computation*, 121:256–285.

[Freund and Schapire, 1996] Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *In Proceedings of the 13th International Conference on Machine Learning*, pages 148–156, Bari, Italy.

[Fukushima, 1979] Fukushima, K. (1979). Neural network model for a mechanism of pattern recognition unnafected by shifts in position – Neocognitron. *Electronics and Communications*, 62:11–18.

[Gauss, 1821] Gauss, C. F. (1821). *Theoria combinationis observationum erroribus minimis obnoxiae.*

[Gibbs, 1997] Gibbs, M. N. (1997). *Bayesian Gaussian processes for regression and classification.* PhD thesis, University of Cambridge, Cambridge, UK.

[Girolami and Calderhead, 2011] Girolami, M. and Calderhead, B. (2011). Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73:123–214.

[Goldberg et al., 1998] Goldberg, P. W., Williams, C. K. I., and Bishop, C. M. (1998). Regression with input-dependent noise: A Gaussian process treatment.

In Jordan, M. I., Kearns, M. J., and Solla, S. A., editors, *Advances in Neural Information Processing Systems*, volume 10, pages 493–499. The MIT Press, Cambridge, MA.

[Hansen and Salamon, 1990] Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:993–1001.

[Hansen and Lunde, 2005] Hansen, P. R. and Lunde, A. (2005). A forecast comparison of volatility models: Does anything beat a GARCH (1, 1)? *Journal of Applied Econometrics*, 20:873–889.

[Harrison and Rubinfeld, 1978] Harrison, D. and Rubinfeld, D. L. (1978). Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5:81–102.

[Harville, 1997] Harville, D. A. (1997). *Matrix Algebra from a Statistician's Perspective*. Springer. New York, NY.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, NY.

[Hastings, 1970] Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109.

[Haykin, 2007] Haykin, S. (2007). *Neural Networks. A Comprehensive Foundation (3rd edition)*. Prentice-Hall, Upper Saddle River, NJ.

[Hebb, 1949] Hebb, D. O. (1949). *The Organization of Behavior*. Wiley, New York, NY.

[Hinkley, 1969] Hinkley, D. V. (1969). On the ratio of two correlated normal random variables. *Biometrika*, 56:635–639.

[Hinton et al., 2006] Hinton, G. E., Osindero, S., and Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Compututation*, 18:1527–1554.

[Hinton and Salakhutdinov, 2006] Hinton, G. E. and Salakhutdinov, R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313:504–507.

[Hinton and Sejnowski, 1986] Hinton, G. E. and Sejnowski, T. J. (1986). Learning and relearning in boltzmann machines. In Rumelhart, D. E. and McClelland, J. L., editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, pages 282–317. MIT Press, Cambridge, MA.

[Hollander et al., 1999] Hollander, M., Wolfe, D. A., and Chicken, E. (1999). *Nonparametric Statistical Methods*. Wiley, Hoboken, NJ.

[Hornik et al., 1989] Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366.

[Jacobs et al., 1991] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3:79–87.

[Jordan and Jacobs, 1994] Jordan, M. I. and Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214.

[Journel and Huijbregts, 1978] Journel, A. G. and Huijbregts, C. J. (1978). *Mining Geostatistics*. Academic Press, New York, NY.

[Kahneman, 2011] Kahneman, D. (2011). *Thinking, Fast and Slow*. Farrar, Straus and Giroux, New York, NY.

[Karush, 1939] Karush, W. (1939). Minima of Functions of Several Variables with Inequalities as Side Constraints. Master's thesis, Deptartment of Mathematics, University of Chicago, IL.

[Kersting et al., 2007] Kersting, K., Plagemann, C., Pfaff, P., and Burgard, W. (2007). Most likely heteroscedastic Gaussian process regression. In Ghahramani,

Z., editor, *Proceedings of the 24th International Conference on Machine Learning*, pages 393–400, Corvallis, OR. Omnipress.

[Kimeldorf and Wahba, 1971] Kimeldorf, G. S. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *J. Mathematical Analysis and Applications*, 33:82–95.

[Kolmogorov, 1939] Kolmogorov, A. N. (1939). Sur l'interpolation et extrapolation des suites stationnaires. *Comptes Rendues Academie Sciences*, 208:2043–2045.

[Kuhn and Tucker, 1951] Kuhn, H. W. and Tucker, A. W. (1951). Nonlinear programming. In Neyman, J., editor, *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492. University of California Press, Berkeley, CA.

[Kuncheva, 2004] Kuncheva, L. I. (2004). *Combining Pattern Classifiers: Methods and Algorithms*. Wiley, Hoboken, NJ.

[Lázaro-Gredilla and Titsias, 2011] Lázaro-Gredilla, M. and Titsias, M. K. (2011). Variational heteroscedastic Gaussian process regression. In Getoor, L. and Scheffer, T., editors, *Proceedings of the 28th International Conference on Machine Learning*, pages 841–848, Bellevue, WA. ACM.

[Le et al., 2005] Le, Q. V., Smola, A. J., and Canu, S. (2005). Heteroscedastic Gaussian process regression. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 489–496, Bonn, Germany. ACM.

[LeCun, 1985] LeCun, Y. (1985). Une procédure d'apprentissage pour réseau à seuil asymétrique. In *Proceedings of Cognitiva*, volume 85, pages 599–604, Paris.

[LeCun et al., 1989] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551.

[Liu, 2001] Liu, J. (2001). *Monte Carlo Strategies in Scientific Computing*. Springer, New York, NY.

[Lütkepohl, 1996] Lütkepohl, H. (1996). *Handbook of Matrices*. Wiley. West Sussex, UK.

[Matheron, 1973] Matheron, G. (1973). The intrinsic random functions and their applications. *Advances in Applied Probability*, 5:439–468.

[McCulloch and Pitts, 1943] McCulloch, W. S. and Pitts, W. H. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133.

[McCullough and Renfro, 1998] McCullough, B. D. and Renfro, C. G. (1998). Benchmarks and software standards: A case study of GARCH procedures. *Journal of Economic and Social Measurement*, 25:59–71.

[Michalis and Lázaro-Gredilla, 2013] Michalis, T. and Lázaro-Gredilla, M. (2013). Variational inference for Mahalanobis distance metrics in Gaussian process regression. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 26, pages 279–287. Curran Associates, Inc.

[Minka, 2001] Minka, T. P. (2001). *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA.

[Minsky and Papert, 1969] Minsky, M. L. and Papert, S. A. (1969). *Perceptrons*. MIT Press, Cambridge, MA.

[Muñoz-González et al., 2011] Muñoz-González, L., Lázaro-Gredilla, M., and Figueiras-Vidal, A. R. (2011). Heteroscedastic Gaussian process regression using expectation propagation. In Tan, T., Katagiri, S., Tao, J., Nakamura, A.,

and Larsen, J., editors, *Proceedings of the International Workshop on Machine Learning for Signal Processing*, pages 1–6, Beijing, China. IEEE.

[Muñoz-González et al., 2014a] Muñoz-González, L., Lázaro-Gredilla, M., and Figueiras-Vidal, A. R. (2014a). Divisive Gaussian processes for nonstationary regression. *IEEE Transactions on Neural Networks and Learning Systems*. Accepted for publication.

[Muñoz-González et al., 2014b] Muñoz-González, L., Lázaro-Gredilla, M., and Figueiras-Vidal, A. R. (2014b). Laplace approximation for divisive Gaussian processes for nonstationary regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Submitted.

[Muñoz-González et al., 2014c] Muñoz-González, L., Lázaro-Gredilla, M., and Figueiras-Vidal, A. R. (2014c). Laplace approximation with Gaussian processes for volatility forecasting. In Hansen, L. K., Jensen, S. H., and Larsen, J., editors, *Proceedings of the 4th International Workshop on Cognitive Information Processing*, pages 1–6, Copenhagen. IEEE.

[Murray et al., 2010] Murray, I., Adams, R. P., and MacKay, D. J. C. (2010). Elliptical slice sampling. In *International Conference on Artificial Intelligence and Statistics 13*, volume 9 of Journal of Machine Learning Research, pages 541–548, Sardinia, Italy.

[Myers, 2004] Myers, D. G. (2004). *Intuition: Its Powers and Perils*. Yale University Press, New Haven, CT.

[Neal, 1999] Neal, R. M. (1999). Regression and classification using Gaussian process priors. *Bayesian Statistics*, 6:475–501.

[Neal, 2003] Neal, R. M. (2003). Slice sampling. *Annals of Statistics*, 31:705–741.

[Nix and Weigend, 1995] Nix, D. A. and Weigend, A. S. (1995). Learning local error bars for nonlinear regression. In Tesauro, G., Touretzky, D. S., and Leen, T. K.,

editors, *Advances in Neural Information Processing Systems*, volume 7, pages 489–496. The MIT Press, Cambridge, MA.

[O'Hagan and Kingman, 1978] O'Hagan, A. and Kingman, J. F. C. (1978). Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–42.

[Opper and Winther, 2000] Opper, M. and Winther, O. (2000). Gaussian processes for classification: Mean-field algorithms. *Neural Computation*, 12:2655–2684.

[Parker, 1982] Parker, D. B. (1982). Learning Logic. Technical Report S81-64 F1, Stanford University, CA.

[Penrose et al., 1985] Penrose, K. W., Nelson, A. G., and Fisher, A. G. (1985). Generalized body composition prediction equation for men using simple measurement techniques. *Medicine & Science in Sports & Exercise*, 17:189.

[Poon and Granger, 2003] Poon, S. H. and Granger, C. W. J. (2003). Forecasting volatility in financial markets: A review. *Journal of Economic Literature*, 41:478–539.

[Powell, 1987] Powell, M. J. D. (1987). Radial basis functions for multivariable interpolation: A review. In Mason, J. C. and Cox, M. G., editors, *Algorithms for Approximation*, pages 143–167. Oxford University Press, Oxford, UK.

[Quadrianto et al., 2009] Quadrianto, N., Kersting, K., Reid, M. D., Caetano, T. S., and Buntine, W. L. (2009). Kernel conditional quantile estimation via reduction revisited. In *Proceedings of the 9th International Conference on Data Mining*, pages 938–943, Miami, FL. IEEE.

[Quinlan, 1992] Quinlan, J. R. (1992). Learning with continuous classes. In *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, pages 343–348. Singapore: World Scientific.

[Quinlan, 1993] Quinlan, J. R. (1993). Combining instance-based and model-based learning. In *Proceedings on the 10th International Conference of Machine Learning*, pages 236–243, Amherst, MA. Morgan Kaufmann.

[Quiñonero-Candela et al., 2007] Quiñonero-Candela, J., Snelson, E., and Williams, O. (2007). Sensible Priors for Sparse Bayesian Learning. Technical Report MSR-TR-2007-121, Microsoft Research, Cambridge, UK.

[Rasmussen and Williams, 2006] Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA.

[Rokach, 2010] Rokach, L. (2010). *Pattern Classification Using Ensemble Methods*. World Scientific, Singapore.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408.

[Rumelhart et al., 1986a] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986a). Learning internal representations by error propagation. In Rumelhart, D. E., McClelland, J. L., and Group, P. R., editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, pages 318–362. MIT Press, Cambridge, MA.

[Rumelhart et al., 1986b] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986b). Learning representations by back-propagating errors. *Nature*, 323:533–536.

[Schapire and Freund, 2012] Schapire, R. E. and Freund, Y. (2012). *Boosting: Foundations and Algorithms*. MIT Press, Cambridge, MA.

[Schmidhuber, 2014] Schmidhuber, J. (2014). Deep learning in neural networks: An overview. *arXiv preprintv(arXiv:1404.7828)*.

[Schölkopf and Smola, 2001] Schölkopf, B. and Smola, A. J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, Cambridge, MA.

[Seeger, 2005] Seeger, M. (2005). Expectation Propagation for Exponential Families. Technical Report EPFL-REPORT-161464, Berkeley University, CA.

[Sharkey, 1999] Sharkey, A. J. (1999). *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems.* Springer-Verlag, London, UK.

[Silverman, 1985] Silverman, B. W. (1985). Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society. Series B (Methodological)*, 47:1–52.

[Snelson and Ghahramani, 2006] Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In Weiss, Y., Schölkopf, B., and Platt, J. C., editors, *Advances in Neural Information Processing Systems*, volume 18, pages 1257–1264. MIT Press, Cambridge, MA.

[Stamey et al., 1989] Stamey, T. A., Kabalin, J. N., McNeal, J. E., Johnstone, I. M., Freiha, F., Redwine, E. A., and Yang, N. (1989). Prostate specific antigen in the diagnosis and treatment of adenocarcinoma of the prostate. II: Radical prostatectomy treated patients. *The Journal of Urology*, 141:1076–1083.

[Sutton and Barto, 1998] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning. An Introduction.* MIT Press, Cambridge, MA.

[Tipping, 2001] Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244.

[Titsias, 2009] Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 567–574, Clearwater Beach, FL.

[Tsanas et al., 2010] Tsanas, A., Little, M. A., McSharry, P. E., and Ramig, L. O. (2010). Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests. *IEEE Transactions on Biomedical Engineering*, 57:884–893.

[Turing, 1992] Turing, A. M. (1992). *Intelligent Machinery.* Elsevier, Amsterdam, NY.

[Uhlenbeck and Ornstein, 1930] Uhlenbeck, G. E. and Ornstein, L. S. (1930). On the theory of the Brownian motion. *Physical Review*, 36:823–841.

[Vapnik, 1982] Vapnik, V. (1982). *Estimation of Dependences Based on Empirical Data.* Springer, New York, NY.

[Vapnik, 1999] Vapnik, V. (1999). *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, NY.

[Vincent et al., 2008] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P. A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1096–1103, New York, NY. ACM Press.

[Werbos, 1974] Werbos, P. J. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* PhD thesis, Harvard University, Cambridge, MA.

[Wiener, 1949] Wiener, N. (1949). *Extrapolation, Interpolation, and Smoothing of Stationary Time Series.* MIT Press, Cambridge, MA.

[Williams and Rasmussen, 1996] Williams, C. K. I. and Rasmussen, C. E. (1996). Gaussian processes for regression. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems*, volume 8, pages 514–520. MIT Press, Cambridge, MA.

[Wilson and Ghahramani, 2010] Wilson, A. G. and Ghahramani, Z. (2010). Copula processes. In Jafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A., editors, *Advances in Neural Information Processing Systems*, volume 23, pages 2460–2468. Curran Associates, Inc.

[Wolpert, 1992] Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5:241–259.

[Yeh, 1998] Yeh, I. C. (1998). Modeling of strength of high-performance concrete using artificial neural networks. *Journal of Cement and Concrete Research*, 28:1797–1808.

[Yuan and Wahba, 2004] Yuan, M. and Wahba, G. (2004). Doubly penalized likelihood estimator in heteroscedastic regression. *Statistics and Probability Letters*, 69:11–20.

[Zoeter and Heskes, 2005] Zoeter, O. and Heskes, T. (2005). Gaussian quadrature based expectation propagation. In Robert, G. C. and Ghahramani, Z., editors, *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, pages 445–452. Society for Artificial Intelligence and Statistics.